Functional Approximations to Posterior Densities: A Neural Network Approach to Efficient Sampling

Lennart F. Hoogerheide*, Johan F. Kaashoek & Herman K. van Dijk

Econometric Institute, Erasmus University Rotterdam

Econometric Institute Report 2002-48

February 15, 2003 (revised March 29, 2003)

Abstract

The performance of Monte Carlo integration methods like importance sampling or Markov Chain Monte Carlo procedures greatly depends on the choice of the importance or candidate density. Usually, such a density has to be 'close' to the target density in order to yield numerically accurate results with efficient sampling. Neural networks seem to be natural importance or candidate densities, as they have a universal approximation property and are easy to sample from. That is, conditionally upon the specification of the neural network, sampling can be done either directly or using a Gibbs sampling technique, possibly using auxiliary variables. A key step in the proposed class of methods is the construction of a neural network that approximates the target density accurately. The methods are tested on a set of illustrative models which include a mixture of normal distributions, a Bayesian instrumental variable regression problem with weak instruments and near-identification, and a two-regime growth model for US recessions and expansions. These examples involve experiments with non-standard, non-elliptical posterior distributions. The results indicate the feasibility of the neural network approach.

Keywords: importance sampling, Markov chain Monte Carlo, neural networks, Bayesian inference. **JEL classification:** C11, C15, C45

*Correspondence to: L.F. Hoogerheide, Econometric Institute, Erasmus University Rotterdam, P.O. Box 1738, NL-3000 DR Rotterdam, The Netherlands. E-mail: lhoogerheide@few.eur.nl

1 Introduction

Markov Chain Monte Carlo (MCMC) methods like Metropolis-Hastings (MH) and Gibbs sampling are extensively used in Bayesian analyses of econometric and statistical models. The theory of Markov chain samplers starts with Metropolis et al. (1953) and Hastings (1970). An important technical paper on MCMC methods is due to Tierney (1994). Well known econometric studies are provided by Chib and Greenberg (1996) and Geweke (1999). Indirect independence sampling methods such as importance sampling (IS) have also been successfully applied within Bayesian inference. Importance sampling, see Hammersley and Handscomb (1964), has been introduced in Bayesian inference by Kloek and Van Dijk (1978) and is further developed by Van Dijk and Kloek (1980,1984) and Geweke (1989).

However, in practice, the convergence behavior of Monte Carlo methods is still often uncertain. The complex structure of a model or some extraordinary properties of the data may cause this problem. We mention three cases. First, Hobert and Casella (1996) show that the Gibbs sampler does not converge in the case of a hierarchical linear mixed model if the prior is uniform. The reason is that the posterior of some conditional variance is improper. Similar problems may occur in dynamic panel data models using diffuse priors. A second example of a complex model is a set of equations with a near reduced rank structure for the matrix of coefficients. Then the Hessian of the likelihood function is singular. This may be due to near nonidentifiability or near nonstationarity in econometric models with diffuse priors. We refer to the studies by Schotman and Van Dijk (1991) and Kleibergen and Van Dijk (1994, 1998). Convergence problems of importance sampling with a normal or Student t importance density are described by Van Dijk and Kloek (1984) and Geweke (1989). As a third case we mention a multimodal target density, which one may encounter in mixture processes with a small number of observations around one of the different modes. This may cause problems for all methods. If the MH candidate density is unimodal, with a low probability of drawing candidate values in one of the modes, then this mode may be completely missed, even if the sample size gets very large. In this case importance sampling with a unimodal normal or Student t importance density may yield a sample in which most drawings have a negligible weight and only a few drawings almost completely determine the sampling results.

So, an important problem is the choice of the candidate or importance density, especially when one knows little about the shape of the target density.

In this paper we introduce the class of neural network sampling methods to sample from a target (posterior) distribution that may be multi-modal or skew, or exhibit strong nonlinear correlation among the parameters. That is, a class of methods to sample from non-elliptical distributions.

The basic idea of the neural network sampling algorithms is simple. First, a neural network is constructed that approximates the target density. An important advantage of neural network functions is their 'universal approximation property'. That is, neural network functions can provide approximations of any square integrable function to any desired accuracy, see Gallant and White (1989). As an application of Kolmogorov's general superposition theorem, the neural network approximation property is eluded by Hecht-Nielsen (1987). Proofs concerning neural network approximations for specific configurations can be found in Gallant and White (1989), Hornik et al. (1989), and Leshno et al. (1993). This approximation property implies that the algorithm can handle certain 'strange' target distributions, like multi-modal, extremely skew, strongly correlated or fat-tailed distributions. Second, this neural network is used as an importance function in IS or as a candidate density in MH. Depending on the specification of the neural network, an important advantage of neural network densities is that they are easy to sample from.

The proposed methods are applied on a set of illustrative examples. We start with a mixture of normal densities. Next we perform some experiments with a Bayesian analysis of the instrumental variable regression model. Finally, we explore a switching model with recessions and expansions for the US real GNP growth. Our results indicate that the neural network approach is feasible in cases where a 'standard' MH, Gibbs or IS approach would fail or be extremely slow.

The outline of the paper is as follows. In section 2 we discuss how to construct a neural network approximation to a density, how to sample from a neural network density, and how to use these drawings within the IS or MH algorithm. In section 3 we describe a method yielding estimates of moments of the target distribution without requiring a sampling algorithm. Section 4 shows the feasibility of our approach in a simple example of a mixture of bivariate normal distributions. Section 5 illustrates our algorithms in an example with simulated data in an instrumental variable (IV) regression set up. Section 6 contains an empirical example concerning a switching model for the quarterly growth rate of the real GNP in the USA. Conclusions are given in section 7 and technical details are given in the appendices.

2 Approximating with and sampling from neural networks

Suppose we have a certain distribution with density function $p(x)$ at hand, where $x \in \mathbb{R}^n$. The aim is to investigate some of the characteristics of $p(x)$, for example the mean and covariance matrix of a random vector $X \sim p(x)$. The approach followed in this paper is:

1. Find a neural network approximation $nn : \mathbb{R}^n \to \mathbb{R}$ to the target density $p(x)$.
2. Obtain a sample of random points from the density $nn(x)$.
3. Perform importance sampling or the Metropolis-Hastings algorithm using this sample in order to obtain estimates of the characteristics of $p(x)$.

Consider a (simplified) 4-layer feed-forward neural network with functional form:

$$nn(x) = G_2 \left(cG_1(Ax + b) + d \right) + \delta \tag{1}$$

where A is $H \times n$, b is $H \times 1$, c is $1 \times H$, $d \in \mathbb{R}$ and $\delta \in \mathbb{R}$. The integer H is interpreted as the number of cells in the first hidden layer of the neural network. The second hidden layer contains only one cell.

The vector function $G_1 : \mathbb{R}^H \to \mathbb{R}^H$ is defined by

$$G_1(y) = (g(y_1), \cdots, g(y_H))' \tag{2}$$

where $g : \mathbb{R} \to \mathbb{R}$, the activation function, is a monotonically increasing function taking its values in the interval $[0, 1]$.
The function $G_2 : \mathbb{R} \to \mathbb{R}$ is a monotonically increasing function, not necessarily bounded. In the following sections, two typical specifications of (1) will be used.

Type 1: A standard three-layer feed-forward neural network ($\delta = 0$ and G_2 is the identity $G_2(x) = x$). As activation function g in (2), we take the scaled arctangent function:

$$g(x) = \frac{1}{\pi} \arctan(x) + \frac{1}{2}. \tag{3}$$

The reason for this choice is that this activation function can be analytically integrated infinitely many times. We will show in subsection 2.2.1, that this property makes the neural network, in the role of a density kernel, easy to sample from.

Type 2: A four-layer network with $\delta = 0$ and G_2 the exponential function:

$$G_2(y) = e^y \tag{4}$$

In this case, the activation g in (2) is taken to be a piecewise-linear function, called *plin*:

$$plin(x) = \begin{cases} 0 & x < -1/2 \\ x + 1/2 & -1/2 \leq x \leq 1/2 \\ 1 & x > 1/2 \end{cases} \tag{5}$$

With this activation function, the neural network function can be analytically integrated. We will show in subsection 2.2.2, that this property makes Gibbs sampling possible.

Note that the function G_2 is unbounded but invertible: this specification may be considered as applying a log-transformation to the data, which correspond in our case to density values. It is possible to specify a different functional form in (4), as long as the function is positive valued and analytically integrable, and its primitive is analytically invertible to allow for easy sampling. An example of such a function is the logistic function. See subsection 2.2.2.

Table 1 gives an overview of the reasons for which we have chosen these particular specifications. The implications shown in this table will be clarified in the sequel of this paper.

In the next subsections we will discuss the three steps of our approach: construction of a neural network, sampling from it, and using the sample in IS or MH.

2.1 Constructing a neural network approximation to a density

We suggest the following procedure to obtain a neural network approximation to a certain target density $p(x)$. First we construct a grid of equidistant points x^i $(i = 1, \ldots, N)$ in the bounded region to which we restrict the random variable $X \in \mathbb{R}^n$ to take its values. Then we approximate the target density $p(x)$ with a neural network by minimizing the sum of squared residuals:

$$SSR(A, b, c, d) = \sum_{i=1}^{N} \left(p(x^i) - nn\left(x^i \middle| A, b, c, d \right) \right)^2. \tag{6}$$

Table 1: Motivation of the particular neural network specifications

specification of $nn(x)$	special properties of $nn(x)$		consequences of special properties of $nn(x)$
Type 1	- The activation function g is analytically integrable infinitely many times.	$\Rightarrow$	- Direct sampling from $nn(x)$ is possible. - Analytical expressions exist for the moments of the distribution with density $nn(x)$.
Type 2	- The activation function g is piecewise-linear. - The function G_2 is positive valued and analytically integrable, and its primitive is analytically invertible.	$\Rightarrow$	- Gibbs sampling from $nn(x)$ is possible.
	- The function G_2 is the exponential function.	$\Rightarrow$	- Auxiliary variable Gibbs sampling from $nn(x)$ is possible.

We choose the smallest neural network, i.e. the one with the least hidden cells, that still gives a 'good' approximation to the target distribution. One could define a 'good' approximation as one with a high enough squared correlation R^2. For example, one could require an R^2 of at least 0.80, 0.90 or 0.95, depending on the dimension or the nature of the target density.

After that, we check the squared correlation R^2 between the neural network and the target density for a (much) larger grid than the 'estimation grid'. If this R^2 is also high enough, then we say that the estimation grid is fine enough. In that case the network does not only provide a good approximation to the target density in the points x^i ($i = 1, \ldots, N$) but also in between. Otherwise, we increase the number of grid points N and start all over again. For example, we make the grid twice as fine for one or more elements of X.

This process continues until the grid is fine enough to allow the neural network to 'feel' the shape of the target density accurately.

In the case of our three-layer neural network, we also have to deal with the problem that the neural network function is not automatically non-negative for each x. In order to prevent this we add a penalty term to (6), and check for non-negativity between the grid points x^i ($i = 1, \ldots, N$) afterwards. If $nn(x)$ is negative for some x, we look for its most negative value, and subtract this negative value from the network's constant d. In that way $nn(x)$ becomes non-negative for each x, so that it is a proper density kernel (on the bounded domain to which we restrict it). In our four-layer neural network the exponential function implies that non-negativity is automatically taken care of.

2.2 Sampling from a neural network density

2.2.1 Sampling from a three-layer neural network density

Suppose the joint density kernel of a certain $X \in \mathbb{R}^n$ is given by a standard three-layer feed-forward neural network function with an activation function g that is analytically integrable infinitely many times. Since the neural network function is a linear combination of these activation functions, the neural network function itself is integrable infinitely many times. Hence one can directly sample from the neural network by iteratively drawing the elements X_i ($i = 1, \ldots, n$) in the following way:

$$\begin{array}{c} \text{Draw } x_1 \text{ from } nn(x_1) \\ \text{Draw } x_2 \text{ from } nn(x_2|x_1) \\ \text{Draw } x_3 \text{ from } nn(x_3|x_1, x_2) \\ \vdots \\ \text{Draw } x_n \text{ from } nn(x_n|x_1, x_2, x_3, \cdots, x_{n-1}) \end{array} \tag{7}$$

where $nn(x_1)$, $nn(x_2|x_1)$, $nn(x_3|x_1, x_2)$, etc. are the marginal and conditional neural network densities corresponding to the joint density kernel $nn(x)$. The marginal distribution function $CDF_{nn}(x_1)$:

$$CDF_{nn}(x_1) = \frac{\int_{-\infty}^{x_1} \int_{-\infty}^{\infty} \cdots \int_{-\infty}^{\infty} nn(\tilde{x}_1, x_2, \ldots, x_n) dx_n \cdots dx_2 d\tilde{x}_1}{\int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \cdots \int_{-\infty}^{\infty} nn(\tilde{x}_1, x_2, \ldots, x_n) dx_n \cdots dx_2 d\tilde{x}_1}, \tag{8}$$

and the conditional distribution function $CDF_{nn}(x_2|x_1)$

$$CDF_{nn}(x_2|x_1) = \frac{\int_{-\infty}^{x_2}\int_{-\infty}^{\infty}\cdots\int_{-\infty}^{\infty} nn(x_1,\tilde{x}_2,\ldots,x_n)dx_n\cdots dx_3 d\tilde{x}_2}{\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\cdots\int_{-\infty}^{\infty} nn(x_1,\tilde{x}_2,\ldots,x_n)dx_n\cdots dx_3 d\tilde{x}_2} \tag{9}$$

etc. can be evaluated analytically.

An example of an activation function that can be analytically integrated infinitely many times, is the scaled arctangent function in (3). Some useful integration formulas for this activation function are given in appendices A.1 and A.2.

So, one can easily sample from the density $nn(x_1)$ or $nn(x_2|x_1)$ in formulas (7) by drawing random variables $U_i(i = 1,\ldots,n)$ from the uniform distribution on $[0,1]$ and then finding the scalars $x_i(i = 1,\ldots,n)$ for which $U = CDF_{nn}(x_1)$, $U = CDF_{nn}(x_2|x_1)$, etc. The calculation of $x_i(i = 1,\ldots,n)$ is done numerically with a simple algorithm such as the bisection method.

2.2.2 Sampling from a four-layer neural network density

Suppose the joint density kernel of a certain $X \in \mathbb{R}^n$ is given by the four-layer neural network function in (1) with $G_2(x) = \exp(x)$, G_1 in (2) and the piecewise-linear activation function in (5). It is fairly easy to perform Gibbs sampling from this distribution, as one can divide the domain of each X_i $(i = 1,\ldots,n)$ into a finite number of intervals on which the conditional neural network density is just the exponent of a linear function. Therefore we can analytically integrate the conditional neural network density, and draw from it using the inverse transformation method. Note the three properties of G_2 mentioned below formula (5) are used here explicitly. Details are given in appendix B.1.

The Gibbs sampling procedure consists of iteratively sampling from one-dimensional conditional distributions:

Specify feasible starting values $x^0 = (x_1^0,\cdots,x_n^0)$.

Do for $j = 1,2,\ldots,m$

$$\begin{aligned}
&x_1^{j+1} \text{ from } nn(x_1|x_2^j, x_2^j,\cdots,x_n^j)\\
&x_2^{j+1} \text{ from } nn(x_2|x_1^{j+1}, x_3^j,\cdots,x_n^j)\\
&x_3^{j+1} \text{ from } nn(x_3|x_1^{j+1}, x_2^{j+1}, x_4^j,\cdots,x_n^j)\\
&\qquad\vdots\\
&x_n^{j+1} \text{ from } nn(x_n|x_1^{j+1}, x_2^{j+1}, x_3^{j+1},\cdots,x_{n-1}^{j+1})
\end{aligned} \tag{10}$$

Under certain regularity conditions, the sequence $\{x^0, x^1,\cdots,x^j,\cdots\}$ converges to a sample from the distribution with joint density $nn(x_1,\ldots,x_n)$.

It is also possible to use a different method to draw from a four-layer neural network density: auxiliary variable Gibbs sampling. Using this method, we do not have to restrict ourselves to the piecewise-linear activation function. It allows for well-known activation functions such as the logistic and scaled arctangent functions. Auxiliary variable Gibbs sampling is a Gibbs sampling technique, developed by Damien et al. (1999). The method is based on work of Edwards and Sokal (1988). In this method, latent variables are introduced in an artificial way in order to facilitate drawing from the full set of conditional distributions.

Auxiliary variable Gibbs sampling is possible if the density kernel p can be decomposed as follows:

$$p(x) \propto \pi(x) \prod_{k=1}^{K} l_k(x), \tag{11}$$

where π is a density kernel from which sampling is easy, and l_k $(k = 1, \ldots, K)$ are non-negative functions of $x \in \mathbb{R}^n$. The trick is that a set $U = (U_1, \ldots, U_K)$ of auxiliary variables is introduced such that a kernel of the joint density of X and U is given by:

$$p(x, u) \propto \pi(x) \prod_{k=1}^{K} I\left\{0 < u_k < l_k(x)\right\}. \tag{12}$$

It is easily seen that (11) is a marginal density kernel corresponding to the joint density (12). Therefore one can sample $X \sim p(x)$ by sampling both X and U from (12) and forgetting U.

Kernels from the conditional distributions of X and U are easily obtained from the joint density kernel:

$$p(x|u) \propto \pi(x) I\left\{l_k(x) > u_k, k = 1, \ldots, K\right\} \tag{13}$$

$$p(u|x) \propto \prod_{k=1}^{K} I\left\{0 < u_k < l_k(x)\right\} \tag{14}$$

It follows from (13) and (14) that an iteration of the auxiliary variable Gibbs sampler consists of drawing X from a truncated version of an 'easy' distribution with density kernel π, and sampling U_k $(k = 1, \ldots, K)$ from K independent uniform distributions.

If X is multi-dimensional, it might be difficult or inefficient to sample the whole vector X from the truncated distribution in (13). In that case one may break up X, and sample its components separately, that is, conditionally on the values of U and the other components of X.

Suppose a density kernel of $X \in \mathbb{R}^n$ is given by

$$p(x) = \begin{cases} nn(x) & \text{if } x_i \in [\underline{x}_i, \bar{x}_i] \ \ \forall i = 1, \ldots, n \\ 0 & \text{else} \end{cases} \tag{15}$$

where $[\underline{x}_i, \bar{x}_i]$ is the interval to which X_i $(i = 1, \ldots, n)$ is restricted. This restriction ensures that (15) is a proper density kernel. The function $nn(x)$ is given by:

$$nn(x) = \exp\left(\sum_{h=1}^{H} c_h \, plin\left(\sum_{i=1}^{n} a_{hi} x_i + b_h\right) + d\right), \tag{16}$$

We rewrite $nn(x)$ as

$$\begin{aligned} nn(x) &= \exp\left(\sum_{h=1}^{H} c_h\, plin\left(\sum_{i=1}^{n} a_{hi}x_i + b_h\right) + d\right) \\ &\propto \prod_{h=1}^{H} \exp\left(c_h\, plin\left(\sum_{i=1}^{n} a_{hi}x_i + b_h\right)\right), \end{aligned}$$

so that the density kernel p in (15) can be rewritten as:

$$p(x) = \prod_{i=1}^{n} I\left\{\underline{x}_i < x_i < \bar{x}_i\right\} \prod_{h=1}^{H} \exp\left(c_h\, plin\left(\sum_{i=1}^{n} a_{hi}x_i + b_h\right)\right). \tag{17}$$

It is now easily seen that (17) has the shape of (11) with

$$\pi(x) = \prod_{i=1}^{n} I\left\{\underline{x}_i < x_i < \bar{x}_i\right\}, \tag{18}$$

$$l_h(x) = \exp\left(c_h\, plin\left(\sum_{i=1}^{n} a_{hi}x_i + b_h\right)\right) \quad \text{for } h = 1, \ldots, H. \tag{19}$$

where $\pi(x)$ is the 'easy' density kernel of n independent variables X_i $(i = 1, \ldots, n)$ with distribution $U(\underline{x}_i, \bar{x}_i)$. This means that we can draw from this distribution using auxiliary variable Gibbs sampling. Appendix B.2 shows the technical details; it appears that in this case auxiliary variable Gibbs sampling only requires sampling from uniform distributions, which is done easily and fast.

Note, as indicated before, that it is also possible to perform auxiliary variable Gibbs sampling from a four-layer neural network density with a different activation function. For example, one may use the scaled arctangent function instead of the piecewise-linear function. An advantage of this function is its smoothness, which may facilitate the numerical gradient-based optimization.

2.3 Importance sampling and Metropolis-Hastings

Once we have obtained a sample of random drawings from the neural network density $nn(x)$, we can use this sample in order to estimate those characteristics of the target density $p(x)$ that we are interested in. For this purpose we can use importance sampling and in some cases the Metropolis-Hastings algorithm.

A discussion of importance sampling can be found in Bauwens et al. (1999). Let X be a random variable with density p. Suppose we are interested in the expectation $E(h(X))$ for a certain function $h : \mathbb{R}^n \to \mathbb{R}$. Then the importance sampling (IS) approach to obtain an estimate of $E(h(X))$ is:

Step 1: Draw a sample of y_i's $(i = 1, \ldots, m)$ from a 'candidate distribution' with density q, the so-called importance function.

Step 2: The estimate of $E(h(X))$ is now given by:

$$h_{IS} = \frac{\sum_{i=1}^{n} w(y_i)h(y_i)}{\sum_{i=1}^{n} w(y_i)}, \tag{20}$$

where $w(x) \equiv p(x)/q(x)$ is the so-called weight function.

The Metropolis-Hastings (MH) algorithm was introduced by Metropolis et al. (1953) and generalized by Hastings (1970). The algorithm samples from a time-reversible Markov chain, converging to the target distribution of the random variable $X \in \mathbb{R}^n$ that we are interested in.

The MH algorithm constructs a Markov chain of m random vectors in the following way:

```
Initialization: choose feasible vector $x^0 \in \mathbb{R}^n$.

Do for $j = 1, 2, \ldots, m$

    Obtain $y$ from a 'candidate' density $q(y|x^{j-1})$.
    Compute the 'acceptance probability' $\alpha(x^{j-1}, y)$:

        $\alpha(x^{j-1}, y) \equiv \min\left\{\frac{p(y)q(y|x)}{p(x)q(x|y)}, 1\right\}$

    Obtain $u$ from the uniform distribution on (0,1).
    If $u \leq \alpha(x^{j-1}, y)$ then $x^j = y$ else $x^j = x^{j-1}$.
```

A realized Markov chain can be used in a number of ways. One way is considering all realizations after a certain burn-in period, and using the sample statistics of these realizations as estimates of the characteristics of the distribution of X that we are interested in.

Note that in the case of a four-layer neural network we need Gibbs sampling in order to obtain the sample, so that the consecutive drawings are not independent. In this case it is not efficient to use the Metropolis-Hastings algorithm, since each independent drawing from the candidate density $nn(x)$ would require a whole Gibbs sequence of drawings. The problem is that it is difficult to compute the transition density $q(y|x^{j-1})$, if x^{j-1} and y come from the same Gibbs sequence.

Therefore we have four 'neural network based' algorithms at hand: Neural Network Importance Sampling (NNIS) and Neural Network Metropolis-Hastings (NNMH) in which IS or MH is performed using random vectors that are (directly) drawn from a 3-layer neural network; Gibbs Neural Network Importance Sampling (GiNNIS) and Gibbs with Auxiliary Variables Neural Network Importance Sampling (GiAuVaNNIS) in which IS is performed using random vectors that are drawn from a 4-layer neural network by Gibbs sampling (possibly with auxiliary variables). Table 2 gives an overview.

Table 2: Overview of 'neural network based' sampling algorithms

	Importance sampling	Metropolis-Hastings
3-layer neural network: direct sampling	NNIS	NNMH
4-layer neural network: (auxiliary variable) Gibbs sampling	Gi(AuVa)NNIS	-

3 Analytical expressions for moments of the three-layer neural network distribution

There exist analytical expressions for the moments of the 3-layer neural network distribution with the scaled arctangent activation function, just like the expressions for the marginal and conditional distribution functions that make direct sampling possible. The formulas are derived in appendix A.3. This feature of the 3-layer neural network makes the following algorithm possible if one only wants estimates of certain moments of the target distribution:

Step 1: Construct a 3-layer neural network function $nn(x)$ that gives a good approximation to the target density $p(x)$.

Step 2: Compute the moments of the neural network distribution using the formulas in appendix A.3. These moments provide estimates of those moments of the target density $p(x)$ that one is interested in.

In this case no sampling algorithm like MH or IS is needed. As in this case the neural network output is not 'corrected' by MH or IS, the neural network has to be a very accurate approximation to the target density. Otherwise its moments are inaccurate approximations.

4 Example I: mixture of two bivariate normal distributions

In order to illustrate the neural network sampling algorithms in a simple example, we consider the following bimodal distribution:

$$\begin{pmatrix} X_1 \\ X_2 \end{pmatrix} \sim 0.5\, N\left(\begin{pmatrix} -5 \\ -5 \end{pmatrix}, \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \right) + 0.5\, N\left(\begin{pmatrix} 5 \\ 5 \end{pmatrix}, \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \right) \tag{21}$$

We use our algorithms in order to obtain estimates of the mean and standard deviation of X_1 and X_2, and the correlation coefficient $\rho(X_1, X_2)$.

First, we restrict the variables X_1 and X_2 to the interval [-10,10]. That is, we only consider the region

$$\{(X_1, X_2)| -10 \leq X_1 \leq 10, -10 \leq X_2 \leq 10\}\,. \tag{22}$$

This restriction does not affect our estimates, as the probability mass outside this region is negligible. Then we use the approach described in subsection 2.1 in order to construct a 3-layer neural network approximation to the target density. Since this is only a simple two-dimensional target distribution, we require the squared correlation R^2 between the neural network function and the target density to be at least 0.95. A 41 × 41 grid of equidistant points on the region (22) appears to be large enough: a 3-layer neural network with $H = 48$ hidden cells with an R^2 of 0.985 on the 41 × 41 estimation grid still has an R^2 of 0.985 on an 81 × 81 grid.

We also construct a 4-layer neural network approximation to the target density. We find a 4-layer network with $H = 10$ hidden cells with $R^2 = 0.988$ on the 41×41 estimation grid and $R^2 = 0.984$ on an 81 × 81 grid.

Note the large difference between the sizes of the 3-layer network and the 4-layer network. The 3-layer network requires 5 times as many hidden cells. This suggests that the exponential transformation in the 4-layer network makes it much easier to construct an approximation to the target density.

The contourplots of the 3-layer and 4-layer neural network approximations are given by Figure 1, together with the contourplot of the target density. These contourplots confirm that the neural networks are good approximations to the target density.

After we have constructed neural network approximations, we sample from these networks and use the samples in IS or MH. For each algorithm we construct two samples, and we say that convergence has been achieved if the differences between the two estimated means of X_1 and X_2 are both less than 0.05. The results are in Table 3. Note that the four neural network sampling algorithms - NNIS, NNMH, GiNNIS and GiAuVaNNIS - all yield estimates differing less than 0.05 from the real values. The analytical expressions for the moments of the 3-layer neural network also yield quite good estimates, although not as good as the four neural network sampling algorithms.

NNIS and NNMH require only 50000 drawings, whereas GiNNIS and GiAuVaNNIS require 200000 and 1000000 drawings, respectively. The reason for this is that NNIS

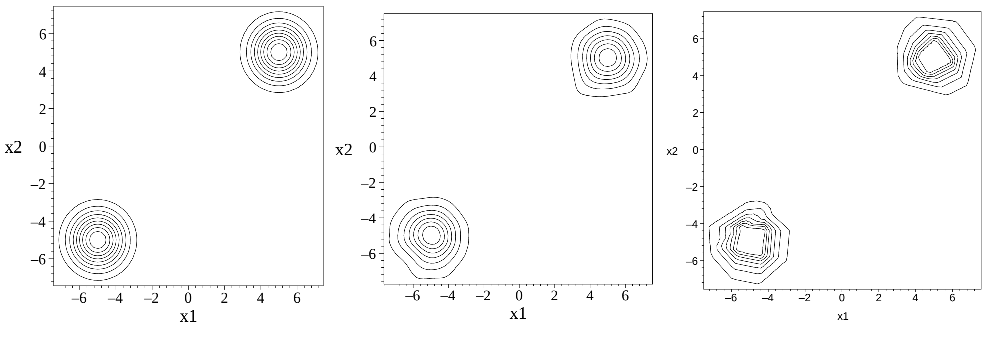

Figure 1: Contourplots: the density of the distribution in (21) (left), its 3-layer neural network approximation (middle), and its 4-layer neural network approximation (right)

and NNMH use a sample of uncorrelated points obtained by direct sampling, whereas GiNNIS and GiAuVaNNIS use Gibbs sequences in which the points are correlated. The first order serial correlations of the Gibbs sequences of X_1's are 0.90 and 0.97 in GiNNIS and GiAuVaNNIS, respectively. Apparently the addition of auxiliary variables increases the serial correlation in the Gibbs sequence, which explains why 1000000 points are needed instead of 200000.

If we look at the computing times (on an AMD Athlon 1400 MHz processor) required for generating the samples, we conclude that the GiNNIS algorithm is the winner in this example. Although in the GiAuVaNNIS method a point is generated faster, the GiNNIS estimates take less time to converge. In other words, in this example GiNNIS appears to be a good trade-off between quality and quantity. The NNIS and NNMH algorithms are relatively slow, as these methods require a numerical method, such as the bisection method, in order to perform the inverse transformation method.

The total weight of the 5% most influential points is below 10% for the three IS algorithms, confirming the high quality of the importance density. The high NNMH acceptance rate of 67% indicates the quality of the neural network as a candidate density.

We now compare the performance of the neural network algorithms with the performance of the Gibbs sampler. Again, we construct two samples, and we say that convergence has been achieved if the differences between the two estimated means of X_1 and X_2 are both less than 0.05. The sampling results are in Table 3. Note the large differences between the neural network algorithms and Gibbs sampling. The scatter plots in Figure 2 of the samples obtained by NNMH and Gibbs sampling reveal the reason for these differences: NNMH yields a fine sample showing the contours of the joint density, whereas the Gibbs sampler completely misses one of the two modes.

We conclude that the neural network approach works very well in this simple example in which the Gibbs sampler fails.

Table 3: Sampling results for the mixture of two bivariate normal distributions

	real	NNIS	NNMH	analytical	GiNNIS	GiAuVa	Gibbs
	values			moments		NNIS	
$E(X_1)$	0	0.024	0.011	0.063	0.035	-0.048	5.014
$E(X_2)$	0	0.022	0.012	0.063	0.040	-0.047	5.012
$\sigma(X_1)$	5.099	5.106	5.102	5.088	5.097	5.099	0.995
$\sigma(X_2)$	5.099	5.103	5.099	5.101	5.104	5.097	0.995
$\rho(X_1, X_2)$	0.962	0.962	0.962	0.968	0.962	0.962	0.003
drawings		50000	50000		200000	1000000	10000
time		568 s	568 s		56 s	172 s	0.1 s
time/draw		11 ms	11 ms		0.28 ms	0.17 ms	0.01 ms
5% weights		8.0%			7.4%	7.4%	
acc. rate			67%				

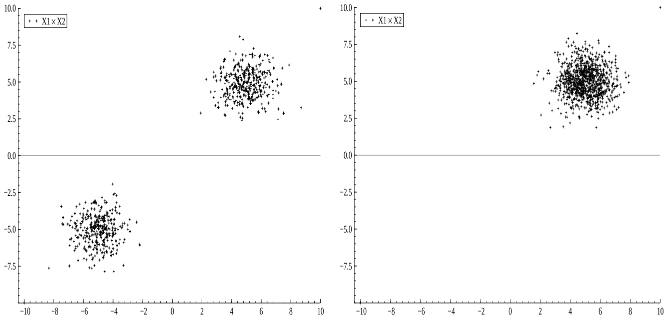

Figure 2: Scatter plots: samples obtained by NNMH (left) and Gibbs sampling (right)

5 Example II: Bayesian analysis of an IV regression

We consider the following equation

$$y_{1t} = y_{2t}\beta + u_{1t} \quad (t = 1, \dots, T) \tag{23}$$

where y_{2t} is a possibly endogenous regressor for which we have

$$y_{2t} = x_t\pi + v_{2t} \quad (t = 1, \dots, T) \tag{24}$$

with $(u_{1t}, v_{2t}) \sim N(0, \Sigma)$ and where x_t is exogenous. We assume a flat prior for the parameters β, π and Σ:

$$p(\beta, \pi, \Sigma) \propto |\Sigma|^{-h/2}, \;\; h > 0 \tag{25}$$

The likelihood function for a sample of size T is

$$L(\beta, \pi, \Sigma | y_1, y_2, x) \propto |\Sigma|^{-T/2} \exp\left[-\frac{1}{2} \mathrm{tr}(\Sigma^{-1} U'U) \right], \tag{26}$$

where $U = (\tilde{u}_1(\beta), \tilde{v}_2(\pi))$ with $\tilde{u}_1(\beta) = y_1 - y_2\beta$ and $\tilde{v}_2(\pi) = y_2 - x\pi$. So, the joint posterior based on the flat prior is

$$p(\beta, \pi, \Sigma | y_1, y_2, x) \propto |\Sigma|^{-(T+h)/2} \exp\left[-\frac{1}{2} \mathrm{tr}(\Sigma^{-1} U'U) \right], \tag{27}$$

Using properties of the inverted Wishart distribution (see Zellner (1971) and Bauwens and Van Dijk (1989)), Σ^{-1} can be analytically integrated out of the joint posterior yielding the following joint posterior for (β, π):

$$p(\beta, \pi | y_1, y_2, x) \propto |U'U|^{-(T+h-3)/2}. \tag{28}$$

In this case a common choice for h is $h = 3$, resulting in the following posterior density

$$p(\beta, \pi | y_1, y_2, x) \propto |U'U|^{-T/2}. \tag{29}$$

In this example we are interested in the (posterior) distribution of the vector (β, π). So, (β, π) plays the role of the random vector X in the previous sections, and $p(\beta, \pi | y_1, y_2, x)$ plays the role of $p(x)$.

Now we simulate $T = 20$ data from the model in (23) and (24) with $\beta = 0$, $\pi = 0.1$, $x_t \sim N(0, 1)$ i.i.d. and

$$\begin{pmatrix} u_{1t} \\ v_{2t} \end{pmatrix} \sim N\left(\begin{pmatrix} 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 1 & 0.99 \\ 0.99 & 1 \end{pmatrix} \right) \quad (t = 1, \dots, T)$$

Note the extremely high correlation $\rho(u_{1t}, v_{2t}) = 0.99$, causing a very strong endogeneity of the regressor y_2 in equation (23). Also note the low value of $\pi = 0.1$, so that x is a weak instrument for y_2. That is, there is ‘weak identification’. We restrict β to the interval $[-5, 5]$ and π to the interval $[-0.25, 0.25]$. Figure 3 shows the contourplot of the posterior density in (29) for our simulated data set.

We use our neural network algorithms to obtain estimates of the posterior means and standard deviations. We find a 3-layer network with $H = 43$ hidden cells with $R^2 = 0.931$ on the 41×41 estimation grid (and $R^2 = 0.930$ on a 81×81 grid), and a 4-layer network with $H = 10$ hidden cells with $R^2 = 0.933$ on the 41×41 estimation grid (and $R^2 = 0.927$ on an 81×81 grid). The contourplots are given by Figure 3. Note that the 4-layer network is much smaller than the 3-layer network, just like in the previous example (see page 12).

For each algorithm we construct two samples, and we say that convergence has been achieved if the differences between the two estimated posterior means of β and π are less than 0.05 and 0.005, respectively. The sampling results are in Table 4.

Note the large differences between the GiNNIS and GiAuVaNNIS results which are based on the same 4-layer neural network approximation. Figure 4 shows the scatter plots of the samples drawn from this 4-layer network using Gibbs sampling and auxiliary variable Gibbs sampling. These scatter plots reveal the reason for the differences: Gibbs sampling yields a fine sample showing the contours of the 4-layer neural network function, whereas auxiliary variable Gibbs sampling completely misses one of the two modes. The first order serial correlations of the Gibbs sequences of π's are 0.85 and 0.92 in GiNNIS and GiAuVaNNIS, respectively. Apparently the addition of auxiliary variables increases the serial correlation in the Gibbs sequence dramatically, so that the auxiliary variable Gibbs sequence does not 'escape' from its top left mode.

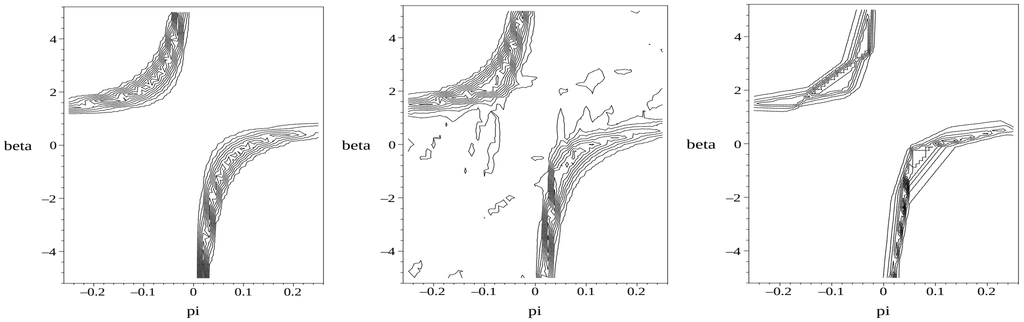

Figure 3: Contourplots: the posterior density in (29) for a simulated data set (left), its 3-layer neural network approximation (middle), and its 4-layer neural network approximation (right)

We now compare the performance of the neural network algorithms with the performance of IS and MH with the maximum likelihood estimator's asymptotic distribution as the candidate density. Recall that the asymptotic distribution of $\hat{\theta}_{ML}$ can be approximated with:

$$N\left(\hat{\theta}_{ML}, \hat{I}\left(\hat{\theta}_{ML}\right)^{-1}\right), \quad \hat{I}\left(\hat{\theta}_{ML}\right) = -\frac{\delta \log L(\hat{\theta}_{ML})}{\delta\theta\delta\theta'} \tag{30}$$

The maximum likelihood estimates are given by Table 5. The estimated correlation is very high: $\hat{\rho}(\hat{\pi}_{ML}, \hat{\beta}_{ML}) = 0.9985$. The sampling results are in Table 4. Note the large differences between the neural network algorithms and IS or MH. The scatter plots in Figure 5 of the NNMH and MH samples reveal the reason for these differences: NNMH yields a fine sample showing the contours of the joint posterior density, whereas MH completely misses one of the two modes.

Table 4: Sampling results for the Bayesian IV regression

		NNIS	NNMH	analytical	GiNNIS	GiAuVa	IS	MH
				moments		NNIS		
π	mean	-0.01	-0.01	-0.01	-0.01	-0.09	-0.06	-0.06
	s.d.	0.10	0.10	0.12	0.11	0.06	0.03	0.03
β	mean	0.65	0.65	0.49	0.67	2.65	2.99	3.05
	s.d.	2.37	2.38	2.49	2.34	0.99	0.93	0.93
drawings		25000	25000		100000	100000	50000	50000
time		261 s	261 s		28 s	17 s	0.05 s	0.05 s
time/draw		10 ms	10 ms		0.28 ms	0.17 ms	0.001 ms	0.001 ms
5% weights		9%			10 %	9 %	41%	
acc. rate			59%					22%

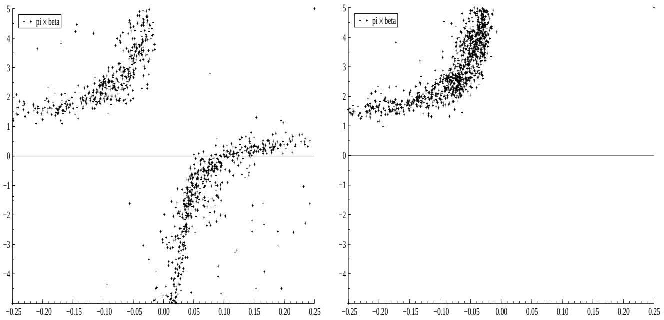

Figure 4: Scatter plots: samples drawn from the 4-layer neural network approximation using Gibbs sampling (left) and auxiliary variable Gibbs sampling (right)

Table 5: Maximum likelihood estimates of the Bayesian IV regression

Parameter:	π	β
MLE:	-0.05	3.36
(std. error)	(0.23)	(11.09)

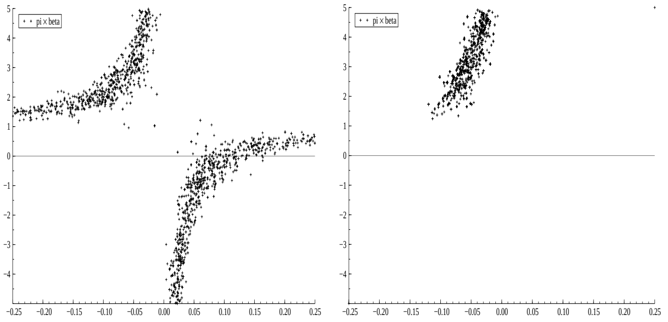

Figure 5: Scatter plots: samples obtained by NNMH (left) and MH (right)

We conclude that NNIS, NNMH and GiNNIS seem to work well in this example (yielding approximately the same estimates). Again, GiNNIS is the fastest among these algorithms. The analytical expressions for the moments of the 3-layer neural network also yield quite good estimates, although not as good as NNIS, NNMH or GiNNIS . In this example GiAuVaNNIS and IS and MH with the maximum likelihood estimator's asymptotic distribution as the candidate distribution do not yield reliable estimates.

6 Example III: Bayesian analysis of a switching model for the quarterly growth rate of the real US GNP

In models for the growth rate of the gross national product one often allows for separate regimes for periods of recession and expansion. One problem that Bayesian analyses of such models may suffer from is the non-convergence of conventional sampling methods. The reason for this is the possible multi-modality of the posterior distribution. We consider the most simple model, a static 2-regime mixture model. In this model the growth rate y_t has two different mean levels:

$$y_t = \begin{cases} \beta_1 + \varepsilon_t \text{ with probability } p \\ \beta_2 + \varepsilon_t \text{ with probability } 1-p \end{cases}, \tag{31}$$

where $\varepsilon_t \sim N(0, \sigma^2)$. For identification we assume that $\beta_1 < \beta_2$, so that β_1 and β_2 can be interpreted as the mean growth rates during recessions and expansions, respectively. The prior densities of the parameters β_1 and β_2 are taken uniform on the set of values for which $\beta_1 < \beta_2$, and zero elsewhere. The prior on p is taken uniform on the interval $[0, 1]$, while for σ the uninformative prior $\pi(\sigma) \propto 1/\sigma$ is used.

The underlying data we consider are the quarterly growth rates of the real US GNP in the period 1959-2001. The data are shown in Figure 6. The maximum likelihood estimates of the parameters are given by Table 6.

We use the neural network algorithms in order to obtain estimates of the posterior mean and standard deviation of β_1, β_2, σ and p. Looking at the graph of the quarterly growth rate and the maximum likelihood estimates, we choose to restrict the parameters to the following intervals: $\beta_1 \in [-3, 1]$, $\beta_2 \in [0.5, 2]$, $\sigma \in [0.5, 1]$ and $p \in [0, 1]$.

We construct a 4-layer neural network approximation to the target density. We find a 4-layer network with $H = 15$ hidden cells with an $R^2 = 0.87$ on the $21 \times 21 \times 11 \times 41$

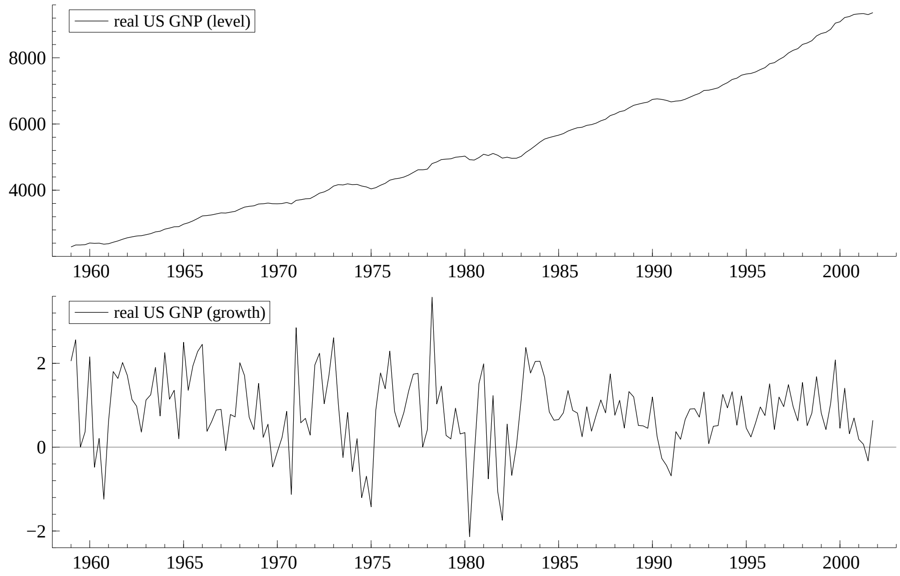

Figure 6: Real GNP of the USA in billions of dollars (above), and its quarterly growth rate in % (below).

Table 6: Maximum likelihood estimates of the 2-regime mixture model

Parameter:	β_1	β_2	σ	p
MLE:	-1.01	0.93	0.79	0.05
(std. error)	(0.51)	(0.08)	(0.06)	(0.04)

estimation grid and $R^2 = 0.86$ on a $41 \times 41 \times 21 \times 81$ grid. As indicated before, a 3-layer neural network approximation requires far more hidden cells than a 4-layer network approximation (see pages 12 and 16). In this case we were not able to construct an accurate 3-layer neural network approximation.

We sample from the 4-layer network using (auxiliary variable) Gibbs sampling, and use the samples in IS. For both GiNNIS and GiAuVaNNIS we construct two samples, and we say that convergence has been achieved if the differences between the estimated posterior means are all less than 0.05. The results are in Table 7. GiNNIS and GiAuVaNNIS estimates of the marginal posterior densities are given by Figure 7 and Figure 8.

We now compare the performance of GiNNIS and GiAuVaNNIS with the performance of IS with the maximum likelihood estimator's asymptotic distribution as the candidate density. The asymptotic distribution of $\hat{\theta}_{ML}$ is approximated with formula (30) on page 16. Again we construct two samples, and we say that convergence has been achieved if the differences between the estimated posterior means are all less than 0.05. The results are in Table 7. IS estimates of the marginal posterior densities are given by Figure 9. Note the large differences, especially in the marginal densities of β_1 and p. The IS estimates indicate a much smaller posterior probability that $\beta_1 \approx 0.8$, and almost zero probability that p exceeds 0.25. Therefore the estimated posterior means of β_1 and p are much smaller according to IS.

The following question remains: which of the results are correct? Or which results are closer to the real values? In order to obtain an answer to this question, we use the fact that we can perform Gibbs sampling from the posterior distribution, if we use the method of data augmentation of Tanner and Wong (1987). Data augmentation is used in order to sample from models with latent variables Z, in which sampling the parameters θ seems very difficult, but sampling θ given Z is straightforward. In this algorithm, the parameters θ are drawn conditionally on the latent variables Z, and the latent variables Z are drawn conditionally on θ. Forgetting the values of Z, this procedure yields a valid Markov chain for the parameters θ. In our model we define the latent variables Z_t $(t = 1, \ldots, T)$ as:

$$Z_t = \begin{cases} 0 & \text{if period } t \text{ is a recession period} \\ 1 & \text{if period } t \text{ is an expansion period} \end{cases} \tag{32}$$

Conditionally on these latent variables Z (and each other), β_1 and β_2 are normally distributed, while σ^2 and p have an inverted gamma and a beta distribution, respectively. Conditionally on the values of the parameters, the latent variables Z_t $(t = 1, \ldots, T)$ have a Bernoulli distribution.

Again we construct two samples, and we say that convergence has been achieved if the differences between the estimated posterior means are all less than 0.05. The results are in Table 7. Data augmentation estimates of the marginal posterior densities are given by Figure 10. Note that the results from the data augmentation algorithm are much closer to the results from GiNNIS and GiAuVaNNIS than to the IS results. This suggests that the results of GiNNIS and GiAuVaNNIS are more reliable than the IS results.

In the GiNNIS method convergence is achieved somewhat faster than in the GiAuVaNNIS algorithm (269 versus 421 seconds on an AMD Athlon 1400 MHz processor). The reason for this is the lower serial correlation in the GiNNIS Gibbs sequence than in the

GiAuVaNNIS Gibbs sequence: for example, for the parameter σ these serial correlations are 0.42 and 0.79, respectively. However, it should also be remarked that the data augmentation algorithm took much less time. Therefore this is just an illustrative example in which the GiNNIS and GiAuVaNNIS algorithms work where another IS algorithm fails.

Table 7: Sampling results for the 2-regime mixture model

	GiNNIS		GiAuVaNNIS		IS		Data Augmentation	
	mean	s.d.	mean	s.d.	mean	s.d.	mean	s.d.
β_1	-0.24	0.84	-0.26	0.83	-0.70	0.68	-0.25	0.86
β_2	0.99	0.17	0.98	0.15	0.93	0.08	1.03	0.29
σ	0.84	0.07	0.83	0.07	0.82	0.06	0.84	0.07
p	0.24	0.27	0.22	0.25	0.07	0.05	0.25	0.29
drawings	400000		800000		400000		400000	
time	269 s		421 s		0.69 s		42 s	
time/draw	0.67 ms		0.53 ms		0.002 ms		0.11 ms	
5% weights	31%		32%		42%			

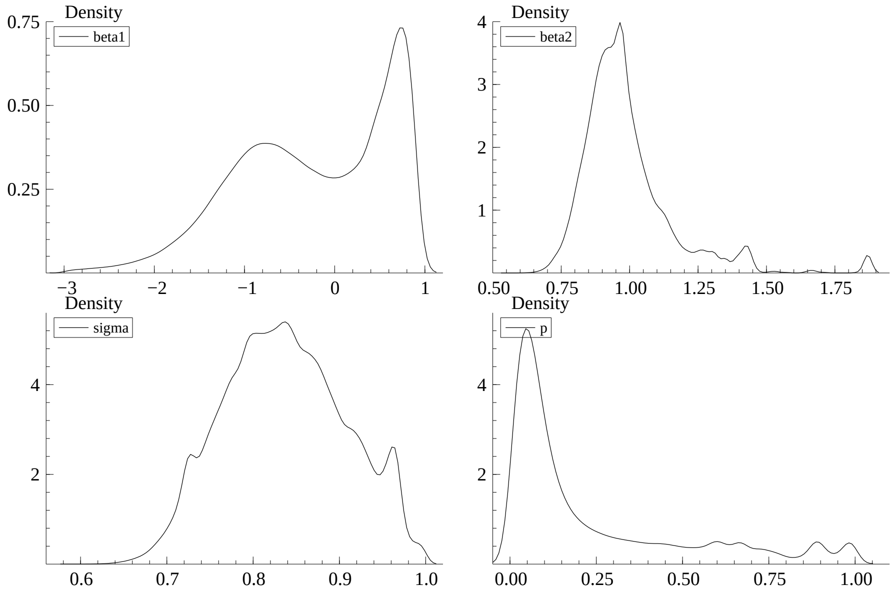

Figure 7: GiNNIS estimates of the marginal densities

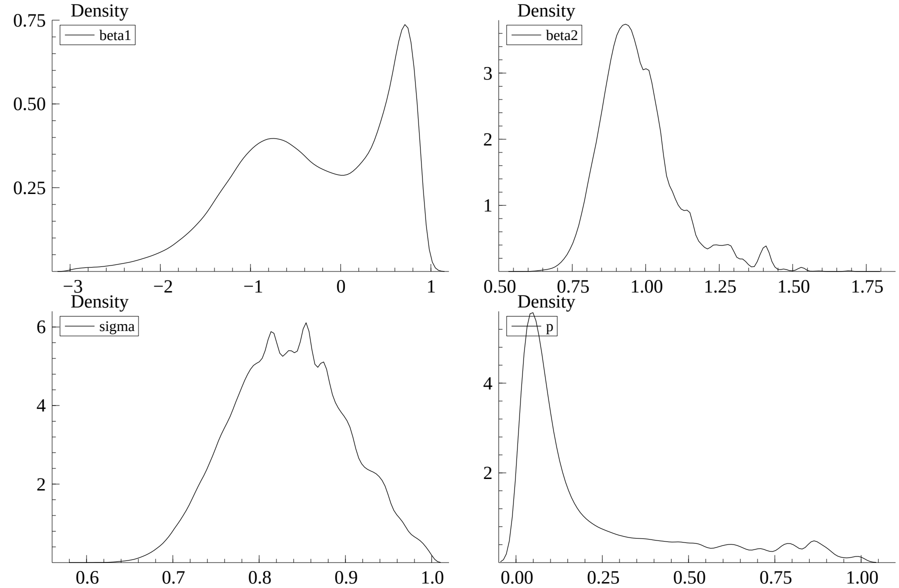

Figure 8: GiAuVaNNIS estimates of the marginal densities

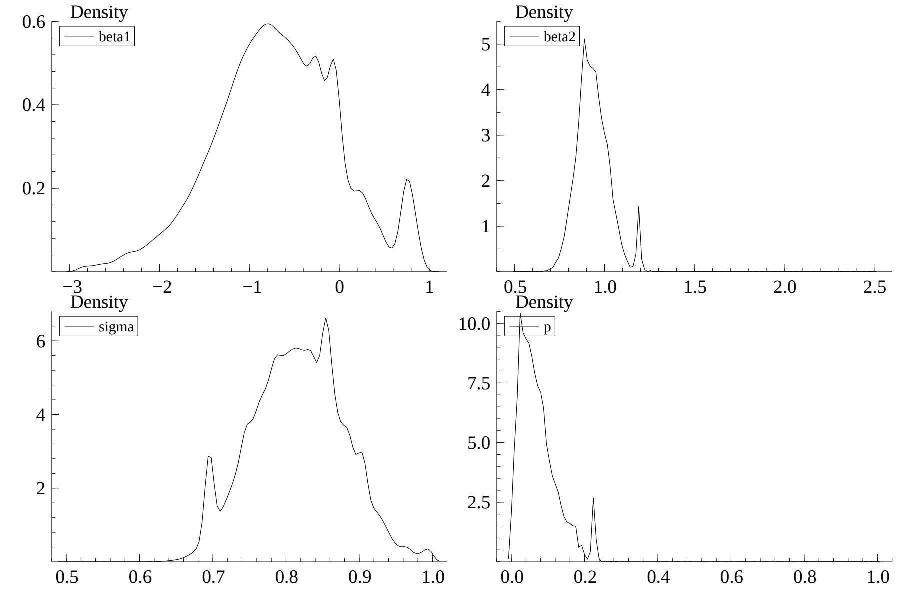

Figure 9: IS estimates of the marginal densities

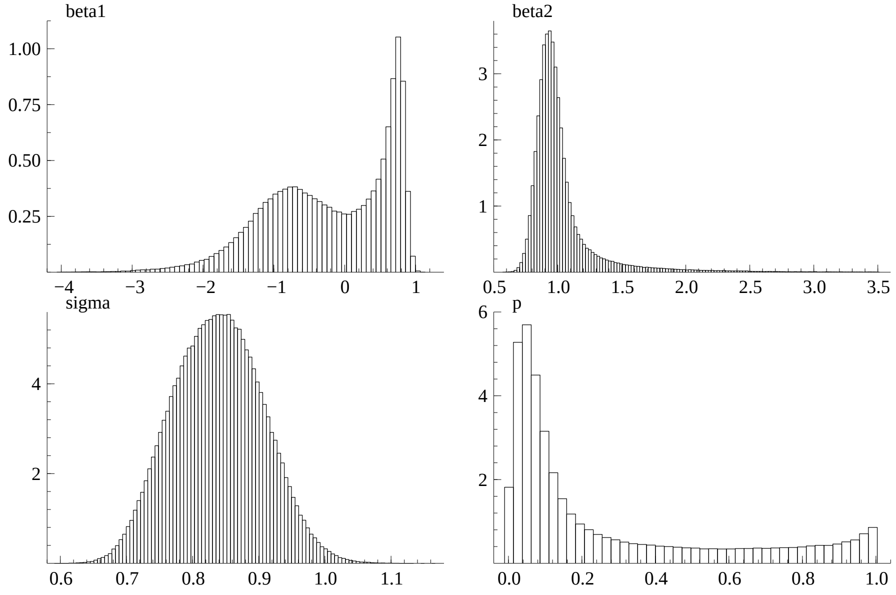

Figure 10: Data augmentation estimates of the marginal densities

7 Conclusion

In this paper we have introduced a class of neural network sampling algorithms. In these algorithms neural network functions are used as an importance or candidate density in importance sampling or the Metropolis-Hastings algorithm. Neural networks are natural importance or candidate densities, as they have a universal approximation property and are easy to sample from. We have shown how to directly sample from a 3-layer neural network, or use Gibbs sampling (possibly with auxiliary variables) to draw from a 4-layer neural network. A key step in the proposed class of methods is the construction of a neural network that approximates the target density accurately. The methods have been tested on a set of illustrative models which include a Bayesian instrumental variable regression problem with weak instruments and near-identification, and a two-regime growth model for US recessions and expansions. In our examples, involving experiments with non-standard, non-elliptical posterior distributions, the GiNNIS algorithm (4-layer neural network, Gibbs sampling) performs the best among the exposed sampling procedures. It is the fastest and moreover the most reliable neural network algorithm, whereas some other algorithms such as the 'usual' Gibbs sampler, MH and IS fail. These results indicate the feasibility and the possible usefulness of the neural network approach.

We end this paper with some remarks on how to extend the proposed techniques. First, one may pursue the construction of well-behaved neural networks with other activation functions which are more smooth than the piecewise-linear one. We noted in section 2 that it is possible to perform auxiliary variable Gibbs sampling from a 4-layer neural network density with a scaled arctangent instead of the piecewise-linear function. One may also investigate the effects of substituting the exponential function in the second hidden layer by a different function such as the logistic function. Second, more experience is needed with empirical econometric models like business cycle models as specified by Hamilton

(1989) and Paap and Van Dijk (2002), stochastic volatility models as given by Shephard (1996), and dynamic panel data models; see Pesaran and Smith (1995). Third, one may, as a first step, transform the posterior density function to a more regular shape. This line of research is recently pursued by, e.g., Bauwens, Bos, Van Dijk and Van Oest (2002) in a class of adaptive direction sampling (ADS) methods. A combination of ADS and neural network sampling may be of interest. Fourth, in practice, one encounters cases where only part of the posterior density is ill-behaved. Then one may combine the neural network approach for the 'difficult part' with a Gibbs sampling approach for the regular part of the model. Finally, in recent work Richard (1998) and Liesenfeld and Richard (2002) constructed an efficient importance sampling technique where the estimation of the parameters of the importance function is done in a sequence of optimization steps. It is also of interest to investigate whether the numerical optimization used for the estimation of the parameters of the neural network approximations can be performed in a more efficient way than proposed in this paper.

A Sampling from a three-layer neural network distribution and computing its moments

Appendix A.1 gives analytical expressions for the integrals of the arctangent function. Appendix A.2 shows how these expressions are used in order to sample from a three-layer neural network distribution. In appendix A.3 these expressions are used to obtain analytical expressions for the moments of a three-layer neural network distribution.

A.1 A simple analytical expression for the integrals of the arctangent function

Theorem A.1: The n-th integral of the arctangent function is given by

$$\begin{aligned} J_n(x) &\equiv \int \cdots \int \arctan(x) dx \cdots dx \\ &= p_n(x) \arctan(x) + q_n(x) \ln(1+x^2) + r_n(x), \end{aligned} \tag{33}$$

where p_n and q_n are polynomials of degree n and $n-1$, respectively:

$$\begin{aligned} p_n(x) &= p_{n,0} + p_{n,1}\, x + \cdots + p_{n,n-1}\, x^{n-1} + p_{n,n}\, x^n \\ q_n(x) &= q_{n,0} + q_{n,1}\, x + \cdots + q_{n,n-1}\, x^{n-1} \end{aligned}$$

The coefficients $p_{n,k}$ $(k = 0, 1, \ldots, n)$ are:

$$p_{n,k} = \begin{cases} \frac{(-1)^{(n-k)/2}}{(n-k)!k!} & \text{if } n-k \text{ is even,} \\ \\ 0 & \text{if } n-k \text{ is odd.} \end{cases} \tag{34}$$

and the coefficients $q_{n,k}$ $(k = 0, 1, \ldots, n-1)$ are given by:

$$q_{n,k} = \begin{cases} \frac{(-1)^{(n-k+1)/2}}{2(n-k)!k!} & \text{if } n-k \text{ is odd,} \\ \\ 0 & \text{if } n-k \text{ is even.} \end{cases} \tag{35}$$

The polynomial r_n (of degree at most $n-1$) plays the role of the integrating constant.

Proof: We will prove this theorem by induction. First, note that for $n = 1$ we have by partial integration with $\arctan(x)$ as the factor to be differentiated:

$$\int \arctan(x) dx = x \arctan(x) - \frac{1}{2} \ln(1+x^2), \tag{36}$$

so that this expression has the shape of formula (33) with $p_{1,0} = 0$, $p_{1,1} = 1$ and $q_{1,0} = -1/2$. It is easily verified that these values correspond with formulas (34) and (35), so that we conclude that for $n = 1$ the proposition holds. Now suppose that our proposition holds for a certain positive integer n. Then we have to show that this implies that the proposition also holds for $n+1$.

First, note that for any positive integer k the integral of $x^k \arctan(x)$ is given by:

$$\int x^k \arctan(x)dx = \frac{1}{k+1}x^{k+1}\arctan(x) - \frac{1}{k+1}\int \frac{x^{k+1}}{1+x^2}dx, \tag{37}$$

which is derived by partial integration with $\arctan(x)$ as the factor to be differentiated. In a similar fashion we have for any positive integer k :

$$\int x^k \ln(1+x^2)dx = \frac{1}{k+1}x^{k+1}\ln(1+x^2) - \frac{2}{k+1}\int \frac{x^{k+2}}{1+x^2}dx. \tag{38}$$

Second, notice that a partial fraction decomposition yields the integral of $x^m/(1+x^2)$ $(m = 0, 1, 2, \ldots)$:

$$\begin{aligned} &\int \frac{x^m}{1+x^2}dx = \\ &= \begin{cases} (-1)^{m/2}\arctan(x) + \sum_{i=0}^{(m-2)/2} \frac{(-1)^i}{m-1-2i}\, x^{m-1-2i} & \text{if } m \text{ is even,} \\ (-1)^{(m-1)/2}\frac{\ln(1+x^2)}{2} + \sum_{i=0}^{(m-3)/2} \frac{(-1)^i}{m-1-2i}\, x^{m-1-2i} & \text{if } m \text{ is odd.} \end{cases} \end{aligned} \tag{39}$$

We may omit the polynomials in (39), since these would eventually be absorbed by the irrelevant polynomial r_n in formula (33), anyway. So, it follows from (37) and (39) that we may use the following equality

$$\begin{aligned} &\int x^k \arctan(x)dx = \\ &= \begin{cases} \frac{1}{k+1}x^{k+1}\arctan(x) - \frac{(-1)^{(k+1)/2}}{k+1}\arctan(x) & \text{if } k \text{ is odd,} \\ \frac{1}{k+1}x^{k+1}\arctan(x) - \frac{(-1)^{k/2}}{2(k+1)}\ln(1+x^2) & \text{if } k \text{ is even.} \end{cases} \end{aligned} \tag{40}$$

In a similar fashion it follows from (38) and (39) that we may use the equality:

$$\begin{aligned} &\int x^k \ln(1+x^2)dx = \\ &= \begin{cases} \frac{1}{k+1}x^{k+1}\ln(1+x^2) - \frac{2(-1)^{(k+2)/2}}{k+1}\arctan(x) & \text{if } k \text{ is even,} \\ \frac{1}{k+1}x^{k+1}\ln(1+x^2) - \frac{(-1)^{(k+1)/2}}{(k+1)}\ln(1+x^2) & \text{if } k \text{ is odd.} \end{cases} \end{aligned} \tag{41}$$

The induction assumption is that for a certain n it holds that:

$$\begin{aligned} J_n(x) &= (p_{n,0} + p_{n,1}\,x + \ldots + p_{n,n}\,x^n)\arctan(x) \\ &\quad + \left(q_{n,0} + q_{n,1}\,x + \ldots + q_{n,n-1}\,x^{n-1}\right)\ln(1+x^2) \end{aligned} \tag{42}$$

where the coefficients $p_{n,k}$ $(k = 0, 1, \ldots, n)$ and $q_{n,k}$ $(k = 0, 1, \ldots, n-1)$ are given by formulas (34) and (35). It follows from (40), (41) and (42) that:

$$\begin{aligned}
J_{n+1}(x) &= \int J_n(x)dx \\
&= \left(p_{n+1,0} + p_{n,0}\,x + \frac{p_{n,1}}{2}\,x^2 + \ldots + \frac{p_{n,n}}{n+1}\,x^{n+1}\right)\arctan(x) \\
&\quad + \left(q_{n+1,0} + q_{n,0}\,x + \frac{q_{n,1}}{2}\,x^2 + \ldots + \frac{q_{n,n-1}}{n}\,x^{n}\right)\ln(1+x^2)
\end{aligned}$$

Note that $J_{n+1}(x)$ has the shape of formula (33) with

$$\begin{aligned}
p_{n+1,k} &= p_{n,k-1}/k \qquad (k = 1, \ldots, n+1), && (43)\\
q_{n+1,k} &= q_{n,k-1}/k \qquad (k = 1, \ldots, n). && (44)
\end{aligned}$$

It follows from (43), (44) and the induction assumption that for $k = 1, \ldots, n+1$ we have:

$$p_{n+1,k} = \begin{cases} \frac{(-1)^{(n-k+1)/2}}{(n-k+1)!(k-1)!k} = \frac{(-1)^{(n+1-k)/2}}{(n+1-k)!k!} & \text{if } n+1-k \text{ is even,} \\ 0 & \text{if } n+1-k \text{ is odd.} \end{cases} \qquad (45)$$

and for $k = 1, \ldots, n$:

$$q_{n+1,k} = \begin{cases} \frac{(-1)^{(n-k+2)/2}}{2(n-k+1)!(k-1)!k} = \frac{(-1)^{(n+1-k+1)/2}}{2(n+1-k)!k!} & \text{if } n+1-k \text{ is odd,} \\ 0 & \text{if } n+1-k \text{ is even.} \end{cases} \qquad (46)$$

Notice that formulas (45) and (46) are just (34) and (35) with $n+1$ instead of n, so that we have proved the correctness of the formula for $k \geq 1$. Now we only have to prove that $p_{n+1,0}$ and $q_{n+1,0}$ are also given by formulas (34) and (35). From (40) and (41) we have:

$$\begin{aligned}
p_{n+1,0} &= \sum_{\{k|1\leq k\leq n;k \text{ odd}\}} -\frac{(-1)^{(k+1)/2}}{k+1} p_{n,k} \\
&\quad + \sum_{\{k|0\leq k\leq n-1;k \text{ even}\}} -\frac{2(-1)^{(k+2)/2}}{k+1} q_{n,k}. \qquad (47)
\end{aligned}$$

Suppose that n is even. Then we have that $n-k$ is odd when k is odd, and $n-k$ is even when k is even. This means that in this case all $p_{n,k}$'s and $q_{n,k}$'s in the two summations of (47) are equal to zero. So, $p_{n+1,0} = 0$ if n is even. If n is odd, we have:

$$\begin{aligned}
p_{n+1,0} &= \sum_{\{k|1\leq k\leq n;k \text{ odd}\}} -\frac{(-1)^{(n+1)/2}}{(n-k)!(k+1)!} \\
&\quad + \sum_{\{k|0\leq k\leq n-1;k \text{ even}\}} -\frac{(-1)^{(n+3)/2}}{(n-k)!(k+1)!},
\end{aligned}$$

which can be rewritten as:

$$\begin{aligned} p_{n+1,0} &= (-1)^{(n+1)/2}\sum_{k=0}^{n}\frac{(-1)^k}{(n-k)!(k+1)!} \\ &= \frac{(-1)^{(n+1)/2}}{(n+1)!}\sum_{k=0}^{n}(-1)^k\left(\begin{array}{c} n+1 \\ k+1 \end{array}\right) = \frac{(-1)^{(n+1)/2}}{(n+1)!}. \end{aligned} \tag{48}$$

The last equality of (48) follows from the fact that:

$$\begin{aligned} \sum_{k=0}^{n}(-1)^k\left(\begin{array}{c} n+1 \\ k+1 \end{array}\right) &= -\sum_{l=1}^{n+1}(-1)^l\left(\begin{array}{c} n+1 \\ l \end{array}\right) \\ &= -\left(\sum_{l=0}^{n+1}(-1)^l\left(\begin{array}{c} n+1 \\ l \end{array}\right) - 1\right) = 1, \end{aligned} \tag{49}$$

which follows from Newton's binomium:

$$\sum_{l=0}^{n+1}(-1)^l\left(\begin{array}{c} n+1 \\ l \end{array}\right) = \sum_{l=0}^{n+1}\left(\begin{array}{c} n+1 \\ l \end{array}\right)(-1)^l\,1^{n+1-l} = (-1+1)^{n+1} = 0.$$

In a similar fashion it follows that

$$\begin{aligned} q_{n+1,0} &= \sum_{\{k|1\leq k\leq n;k \text{ even}\}} -\frac{1}{2}\frac{(-1)^{k/2}}{k+1}p_{n,k} \\ &+ \sum_{\{k|0\leq k\leq n-1;k \text{ odd}\}} -\frac{(-1)^{(k+1)/2}}{k+1}q_{n,k}. \end{aligned} \tag{50}$$

Suppose that n is odd. Then we have that $n-k$ is odd when k is even, and $n-k$ is even when k is odd. This means that in this case all $p_{n,k}$'s and $q_{n,k}$'s in the two summations of (50) are equal to zero. So, $q_{n+1,0} = 0$ if n is odd. If n is even, we have:

$$\begin{aligned} q_{n+1,0} &= \sum_{\{k|1\leq k\leq n;k \text{ even}\}} -\frac{1}{2}\frac{(-1)^{n/2}}{(n-k)!(k+1)!} \\ &+ \sum_{\{k|0\leq k\leq n-1;k \text{ odd}\}} -\frac{1}{2}\frac{(-1)^{(n+2)/2}}{(n-k)!(k+1)!}, \end{aligned}$$

which can be rewritten as:

$$\begin{aligned} q_{n+1,0} &= \frac{(-1)^{(n+2)/2}}{2}\sum_{k=0}^{n}\frac{(-1)^k}{(n-k)!(k+1)!} \\ &= \frac{(-1)^{(n+2)/2}}{2(n+1)!}\sum_{k=0}^{n}(-1)^k\left(\begin{array}{c} n+1 \\ k+1 \end{array}\right) = \frac{(-1)^{(n+2)/2}}{2(n+1)!}, \end{aligned} \tag{51}$$

where the last equality follows from formula (49). From (48) and (51) and the fact that $p_{n+1,0} = 0$ if $n+1$ is odd, and $q_{n+1,0} = 0$ if $n+1$ is even, we conclude that $p_{n+1,0}$ and $q_{n+1,0}$ are also given by formulas (34) and (35), so that we have proved the theorem by induction. □

A.2 The marginal and conditional distribution functions corresponding to a three-layer neural network density

Suppose the random vector $X = (X_1, \ldots, X_n)'$ has the following density $p(x_1, \ldots, x_n)$:

$$p(x_1,\ldots,x_n) = \begin{cases} nn(x_1,\ldots,x_n) & \text{if } \underline{x}_i \leq x_i \leq \bar{x}_i \quad \forall\, i = 1,\ldots,n \\ \\ 0 & \text{else} \end{cases} \tag{52}$$

where $[\underline{x}_i, \bar{x}_i]$ is the interval to which the variable x_i $(i = 1, 2, \ldots, n)$ is restricted. Suppose the function $nn(x_1, \ldots, x_n)$ is given by:

$$nn(x_1,\ldots,x_n) = \sum_{h=1}^{H} c_h\, g(a_{h1}x_1 + \ldots + a_{hn}x_n + b_h) + d \tag{53}$$

with activation function:

$$g(x) = \frac{1}{\pi}\arctan(x) + \frac{1}{2}. \tag{54}$$

Then $nn(x_1, \ldots, x_n)$ can be rewritten as:

$$\begin{aligned} nn(x_1,\ldots,x_n) &= \sum_{h=1}^{H} \frac{c_h}{\pi}\arctan(a_{h1}x_1 + \ldots + a_{hn}x_n + b_h) + \frac{1}{2}\sum_{h=1}^{H} c_h + d \\ &= \sum_{h=1}^{H} \frac{c_h}{\pi}\arctan(a_h' x + b_h) + \frac{1}{2}\sum_{h=1}^{H} c_h + d \end{aligned}$$

Now the cumulative distribution function of X is given by:

$$\begin{aligned} CDF_X(\tilde{x}_1,\ldots,\tilde{x}_n) &= \int_{\underline{x}_n}^{\tilde{x}_n}\cdots\int_{\underline{x}_2}^{\tilde{x}_2}\int_{\underline{x}_1}^{\tilde{x}_1} nn(x_1,\ldots,x_n)dx_1dx_2\cdots dx_n \\ &= \sum_{h=1}^{H}\frac{c_h}{\pi}\int_{\underline{x}_n}^{\tilde{x}_n}\cdots\int_{\underline{x}_2}^{\tilde{x}_2}\int_{\underline{x}_1}^{\tilde{x}_1}\arctan(a_h'x + b_h)dx_1dx_2\cdots dx_n \\ &\quad + \left(\frac{1}{2}\sum_{h=1}^{H} c_h + d\right)x_1x_2\cdots x_n. \end{aligned} \tag{55}$$

Using the fact that $dx_1 = d(a_h'x + b_h)/a_{h1}$ (for constant values of $x_2, \ldots, x_n$), we make the following change of variables:

$$\begin{aligned} &\int_{\underline{x}_1}^{\tilde{x}_1}\arctan(a_h'x + b_h)dx_1 = \\ = \; & \frac{1}{a_{h1}}\int_{a_{h1}\underline{x}_1 + a'_{h,-1}x_{-1} + b_h}^{a_{h1}\tilde{x}_1 + a'_{h,-1}x_{-1} + b_h}\arctan(a_h'x + b_h)d(a_h'x + b_h) \\ = \; & \frac{1}{a_{h1}}\left[J_1(a_{h1}\tilde{x}_1 + a'_{h,-1}x_{-1} + b_h) - J_1(a_{h1}\underline{x}_1 + a'_{h,-1}x_{-1} + b_h)\right], \end{aligned}$$

where we define $a_{h,-1} = (a_{h2}, \ldots, a_{hn})'$ and $x_{-1} = (x_2, \ldots, x_n)'$. In a similar fashion we derive:

$$\begin{aligned}
&\int_{\underline{x}_2}^{\tilde{x}_2} \int_{\underline{x}_1}^{\tilde{x}_1} \arctan(a_h' x + b_h) dx_1 dx_2 = \\
= \quad & \frac{1}{a_{h1}} \left[\int_{\underline{x}_2}^{\tilde{x}_2} J_1(a_{h1}\tilde{x}_1 + a_{h2}x_2 + a_{h,-12}' x_{-12} + b_h) dx_2 \right. \\
& \left. - \int_{\underline{x}_2}^{\tilde{x}_2} J_1(a_{h1}\underline{x}_1 + a_{h2}x_2 + a_{h,-12}' x_{-12} + b_h) dx_2 \right] \\
= \quad & \frac{1}{a_{h1}a_{h2}} \left[J_2(a_{h1}\tilde{x}_1 + a_{h2}\tilde{x}_2 + a_{h,-12}' x_{-12} + b_h) \right. \\
& - J_2(a_{h1}\tilde{x}_1 + a_{h2}\underline{x}_2 + a_{h,-12}' x_{-12} + b_h) \\
& - J_2(a_{h1}\underline{x}_1 + a_{h2}\tilde{x}_2 + a_{h,-12}' x_{-12} + b_h) \\
& \left. + J_2(a_{h1}\underline{x}_1 + a_{h2}\underline{x}_2 + a_{h,-12}' x_{-12} + b_h) \right],
\end{aligned}$$

where we define $a_{h,-12} = (a_{h3}, \ldots, a_{hn})'$ and $x_{-12} = (x_3, \ldots, x_n)'$. If we continue in this way, we obtain the following formula:

$$\begin{aligned}
\int_{\underline{x}_n}^{\tilde{x}_n} \cdots \int_{\underline{x}_2}^{\tilde{x}_2} \int_{\underline{x}_1}^{\tilde{x}_1} \arctan(a_h' x + b_h) dx_1 dx_2 \cdots dx_n = \frac{1}{a_{h1}a_{h2} \cdots a_{hn}} \times \\
\times \sum_{D_1=0}^{1} \cdots \sum_{D_n=0}^{1} (-1)^{D_1+D_2+\cdots+D_n} J_n(a_{h1}x_{1,D_1} + \cdots + a_{hn}x_{n,D_n} + b_h) \qquad (56)
\end{aligned}$$

where we define $x_{i,0} = \tilde{x}_i$ and $x_{i,1} = \underline{x}_i$ $(i = 1, 2, \ldots, n)$, the upper and lower bounds of the integration intervals. The primitive $J_n(x)$ is given by Theorem A.1 in appendix A.1. Substituting (56) into (55) yields:

$$\begin{aligned}
CDF_x(\tilde{x}_1, \ldots, \tilde{x}_n) = \left(\frac{1}{2} \sum_{h=1}^{H} c_h + d \right) x_1 x_2 \cdots x_n + \\
+ \sum_{h=1}^{H} \frac{c_h}{\pi a_{h1}a_{h2} \cdots a_{hn}} \sum_{D_1=0}^{1} \cdots \sum_{D_n=0}^{1} (-1)^{D_1+\cdots+D_n} J_n \left(\sum_{i=1}^{n} a_{hi} x_{i,D_i} + b_h \right). \qquad (57)
\end{aligned}$$

The marginal distribution functions $CDF(x_j)$ $(j = 1, \ldots, n)$ are now obtained by taking $\tilde{x}_i = \bar{x}_i$ $\forall i = 1, \ldots, n; i \neq j$:

$$CDF_{X_j}(x_j) = CDF_x(\bar{x}_1, \ldots, \bar{x}_{j-1}, x_j, \bar{x}_{j+1}, \ldots, \bar{x}_n). \qquad (58)$$

The conditional CDF of x_j given $x_{j+1}, \ldots, x_n$ is given by:

$$CDF(\tilde{x}_j | x_{j+1}, \ldots, x_n) = \frac{\int_{\underline{x}_j}^{\tilde{x}_j} \int_{\underline{x}_{j-1}}^{\bar{x}_{j-1}} \cdots \int_{\underline{x}_1}^{\bar{x}_1} nn(x_1, \ldots, x_n) dx_1 \cdots dx_{j-1} dx_j}{\int_{\underline{x}_j}^{\bar{x}_j} \int_{\underline{x}_{j-1}}^{\bar{x}_{j-1}} \cdots \int_{\underline{x}_1}^{\bar{x}_1} nn(x_1, \ldots, x_n) dx_1 \cdots dx_{j-1} dx_j},$$

for $j = 1, 2, \ldots, n-1$, where

$$\begin{aligned}
&\int_{\underline{x}_j}^{\tilde{x}_j}\cdots\int_{\underline{x}_2}^{\tilde{x}_2}\int_{\underline{x}_1}^{\tilde{x}_1} nn(x_1, \ldots, x_n)dx_1dx_2\cdots dx_j = \\
= \;& \sum_{h=1}^{H}\frac{c_h}{\pi}\int_{\underline{x}_n}^{\tilde{x}_j}\cdots\int_{\underline{x}_2}^{\tilde{x}_2}\int_{\underline{x}_1}^{\tilde{x}_1}\arctan(a_h'x + b_h)dx_1dx_2\cdots dx_j \\
& + \left(\frac{1}{2}\sum_{h=1}^{H}c_h + d\right)x_1x_2\cdots x_j \\
= \;& \sum_{h=1}^{H}\frac{c_h}{\pi a_{h1}a_{h2}\cdots a_{hj}}\sum_{D_1=0}^{1}\cdots\sum_{D_j=0}^{1}(-1)^{D_1+\cdots+D_j}\times \qquad (59) \\
& \times J_j\left(\sum_{i=1}^{j}a_{hi}x_{i,D_i} + \sum_{i=j+1}^{n}a_{hi}x_i + b_h\right) + \left(\frac{1}{2}\sum_{h=1}^{H}c_h + d\right)x_1x_2\cdots x_j,
\end{aligned}$$

where we define $x_{i,0} = \tilde{x}_i$ and $x_{i,1} = \underline{x}_i$ $(i = 1, 2, \ldots, j)$.

As we have formulas (57) and (59) indicating explicit expressions for the marginal and conditional distribution functions, it is easy to sample a random vector from a three-layer neural network density with (scaled) arctangent activation function. We can use the inverse transformation method in the following way:

Step 1: Draw n independent U(0,1) variables $U_1, U_2, \ldots, U_n$.

Step 2: Draw X_n from its marginal distribution by computing the value of X_n such that $CDF_{X_n}(X_n) = U_n$.

Step 3: For $j = n-1, n-2, \ldots, 1$ iteratively draw X_j from its conditional distribution on $X_{j+1}, \ldots, X_n$ by computing the value of X_j such that $CDF(X_j|X_{j+1}, \ldots, X_n) = U_j$.

A.3 Analytical expressions for the moments of a three-layer neural network distribution

Suppose the vector $X = (X_1, \ldots, X_n)'$ has the following density $p(x_1, \ldots, x_n)$:

$$p(x_1, \ldots, x_n) = \begin{cases} nn(x_1, \ldots, x_n) & \text{if } \underline{x}_i \leq x_i \leq \bar{x}_i \quad \forall\, i = 1, \ldots, n \\ \\ 0 & \text{else} \end{cases} \qquad (60)$$

where $[\underline{x}_i, \bar{x}_i]$ is the interval to which the variable x_i $(i = 1, 2, \ldots, n)$ is restricted, and where

$$nn(x_1, \ldots, x_n) = \sum_{h=1}^{H}\frac{c_h}{\pi}\arctan(a_h'x + b_h) + \frac{1}{2}\sum_{h=1}^{H}c_h + d. \qquad (61)$$

Then the expectation of X_n^k $(k = 1, 2, \ldots)$ is given by:

$$\begin{aligned}
& E(X_n^k) = \\
= \;& \int_{\underline{x}_n}^{\bar{x}_n} \int_{\underline{x}_{n-1}}^{\bar{x}_{n-1}} \cdots \int_{\underline{x}_1}^{\bar{x}_1} x_n^k \cdot nn(x_1, \ldots, x_n) dx_1 \cdots dx_{n-1} dx_n \\
= \;& \int_{\underline{x}_n}^{\bar{x}_n} x_n^k \left[\int_{\underline{x}_{n-1}}^{\bar{x}_{n-1}} \cdots \int_{\underline{x}_1}^{\bar{x}_1} nn(x_1, \ldots, x_n) dx_1 \cdots dx_{n-1} \right] dx_n \\
= \;& \sum_{h=1}^{H} \frac{c_h}{\pi} \int_{\underline{x}_n}^{\bar{x}_n} x_n^k \left[\int_{\underline{x}_{n-1}}^{\tilde{x}_{n-1}} \cdots \int_{\underline{x}_1}^{\tilde{x}_1} \arctan(a_h' x + b_h) dx_1 \cdots dx_{n-1} \right] dx_n \\
& + \left(\frac{1}{2} \sum_{h=1}^{H} c_h + d \right) \frac{1}{k+1} (\bar{x}_1 - \underline{x}_1) \cdots (\bar{x}_{n-1} - \underline{x}_{n-1})(\bar{x}_n^{k+1} - \underline{x}_n^{k+1}) \\
= \;& \sum_{h=1}^{H} \frac{c_h}{\pi a_{h1} \cdots a_{h,n-1}} \sum_{D_1=0}^{1} \cdots \sum_{D_{n-1}=0}^{1} \Big[(-1)^{D_1 + \cdots + D_{n-1}} \times \\
& \qquad \times \int_{\underline{x}_n}^{\bar{x}_n} x_n^k \, J_{n-1} \left(\sum_{i=1}^{n-1} a_{hi} x_{i,D_i} + a_{hn} x_n + b_h \right) dx_n \Big] \\
& + \left(\frac{1}{2} \sum_{h=1}^{H} c_h + d \right) \frac{1}{k+1} (\bar{x}_1 - \underline{x}_1) \cdots (\bar{x}_{n-1} - \underline{x}_{n-1})(\bar{x}_n^{k+1} - \underline{x}_n^{k+1}),
\end{aligned} \tag{62}$$

where we define $x_{i,0} = \bar{x}_i$ and $x_{i,1} = \underline{x}_i$ $(i = 1, 2, \ldots, n-1)$, the upper and lower bounds of the integration intervals. We now make use of the following theorem:

Theorem A.2: If the n-th integral of a certain function $f : \mathbb{R} \to \mathbb{R}$ is given by $J_n : \mathbb{R} \to \mathbb{R}$, then it holds for $a_h, x \in \mathbb{R}^n$ and $k = 0, 1, 2, \ldots$ that:

$$\begin{aligned}
& \int x_i^k \, J_n(a_h' x + b_h) dx_i = \\
= \;& \frac{1}{a_{hi}} \sum_{m=0}^{k} \left(-\frac{1}{a_{hi}} \right)^m \frac{k!}{(k-m)!} \, x_i^{k-m} \, J_{n+1+m}(a_h' x + b_h).
\end{aligned} \tag{63}$$

Proof: We will prove this theorem by induction with respect to k. First, note that for $k = 0$ we have:

$$\int J_n(a_h' x + b_h) dx_i = \frac{1}{a_{hi}} \int J_n(a_h' x + b_h) d(a_h' x + b_h) = \frac{1}{a_{hi}} J_{n+1}(a_h' x + b_h).$$

It is easily verified that this corresponds to Theorem A.2 for $k = 0$. Now suppose that our proposition holds for a certain nonnegative integer k. Then we have to show that this implies that the proposition also holds for $k + 1$.

Partial integration with x_i^{k+1} as the factor to be differentiated yields:

$$
\begin{aligned}
& \int x_i^{k+1}\, J_n(a_h' x + b_h) dx_i = \\
= \quad & x_i^{k+1} \left[\int J_n(a_h' x + b_h) dx_i \right] - (k+1) \int x_i^k \left[\int J_n(a_h' x + b_h) dx_i \right] dx_i \\
= \quad & x_i^{k+1} \frac{1}{a_{hi}} J_{n+1}(a_h' x + b_h) - \frac{k+1}{a_{hi}} \int x_i^k\, J_{n+1}(a_h' x + b_h) dx_i \qquad (64)
\end{aligned}
$$

The induction assumption is that Theorem A.2 holds for the value k. Using this induction assumption we rewrite the second term of (64) as:

$$
\begin{aligned}
& -\frac{1}{a_{hi}} (k+1) \int x_i^k\, J_{n+1}(a_h' x + b_h) dx_i = \\
= \quad & \frac{1}{a_{hi}} \sum_{m=0}^{k} \left(-\frac{1}{a_{hi}} \right)^{m+1} \frac{(k+1)!}{(k-m)!} x_i^{k-m}\, J_{n+2+m}(a_h' x + b_h) \\
= \quad & \frac{1}{a_{hi}} \sum_{j=1}^{k+1} \left(-\frac{1}{a_{hi}} \right)^{j} \frac{(k+1)!}{(k+1-j)!} x_i^{k+1-j}\, J_{n+1+j}(a_h' x + b_h) \qquad (65)
\end{aligned}
$$

Adding (65) to the first term of (64) yields:

$$
\begin{aligned}
& \int x_i^{k+1}\, J_n(a_h' x + b_h) dx_i = \\
= \quad & \frac{1}{a_{hi}} \sum_{j=0}^{k+1} \left(-\frac{1}{a_{hi}} \right)^{j} \frac{(k+1)!}{(k+1-j)!} x_i^{k+1-j}\, J_{n+1+j}(a_h' x + b_h) \qquad (66)
\end{aligned}
$$

which is just equation (63) with $k+1$ instead of k. We conclude that we have proved Theorem A.2 by induction. □

Substituting equation (63) of Theorem A.2 into (62) yields:

$$
\begin{aligned}
E(X_n^k) \quad = \quad & \sum_{h=1}^{H} \frac{c_h}{\pi a_{h1} \cdots a_{hn}} \sum_{D_1=0}^{1} \cdots \sum_{D_n=0}^{1} \Big[(-1)^{D_1 + \cdots + D_n} \times \\
& \times \sum_{m=0}^{k} \left(-\frac{1}{a_{hn}} \right)^{m} \frac{k!}{(k-m)!} x_n^{k-m}\, J_{n+m}(a_h' x + b_h) \Big] \qquad (67) \\
& + \left(\frac{1}{2} \sum_{h=1}^{H} c_h + d \right) \frac{1}{k+1} (\bar{x}_1 - \underline{x}_1) \cdots (\bar{x}_{n-1} - \underline{x}_{n-1}) (\bar{x}_n^{k+1} - \underline{x}_n^{k+1}).
\end{aligned}
$$

Of course, formula (67) can easily be adjusted to the general case of $E(X_i^k)$ $(i = 1, 2, \ldots, n)$

by taking a_{hi} and x_i instead of a_{hn} and x_n:

$$\begin{aligned} E(X_i^k) \quad = \quad & \sum_{h=1}^{H} \frac{c_h}{\pi a_{h1} \cdots a_{hn}} \sum_{D_1=0}^{1} \cdots \sum_{D_n=0}^{1} \Big[(-1)^{D_1+\cdots+D_n} \times \\ & \times \sum_{m=0}^{k} \left(-\frac{1}{a_{hi}} \right)^m \frac{k!}{(k-m)!} \, x_i^{k-m} \, J_{n+m} \left(\sum_{i=1}^{n} a_{hi} x_{i,D_i} + b_h \right) \Big] \\ & + \left(\frac{1}{2} \sum_{h=1}^{H} c_h + d \right) \frac{1}{k+1} (\bar{x}_i^{k+1} - \underline{x}_i^{k+1}) \prod_{j=1; j \neq i}^{n} (\bar{x}_j - \underline{x}_j) \end{aligned} \tag{68}$$

In a similar fashion it can be derived that $E(X_i X_j)$ $(i, j = 1, 2, \ldots, n; i \neq j)$ is equal to:

$$\begin{aligned} E(X_i X_j) \quad = \quad & \sum_{h=1}^{H} \frac{c_h}{\pi a_{h1} \cdots a_{hn}} \sum_{D_1=0}^{1} \cdots \sum_{D_n=0}^{1} (-1)^{D_1+\cdots+D_n} \times \\ & \times \Bigg[x_i x_j J_n \left(\sum_{i=1}^{n} a_{hi} x_{i,D_i} + b_h \right) \\ & - \frac{a_{hi} x_i + a_{hj} x_j}{a_{hi} a_{hj}} J_{n+1} \left(\sum_{i=1}^{n} a_{hi} x_{i,D_i} + b_h \right) \\ & \quad + \frac{1}{a_{hi} a_{hj}} J_{n+2} \left(\sum_{i=1}^{n} a_{hi} x_{i,D_i} + b_h \right) \Bigg] \\ & + \left(\frac{1}{2} \sum_{h=1}^{H} c_h + d \right) \frac{1}{4} (\bar{x}_i^2 - \underline{x}_i^2)(\bar{x}_j^2 - \underline{x}_j^2) \prod_{k=1; k \neq i,j}^{n} (\bar{x}_k - \underline{x}_k). \end{aligned} \tag{69}$$

Using formulas (68) and (69), one can easily compute statistics of a three-layer feed-forward neural network distribution, such as mean, variance, skewness, kurtosis, covariances and correlations.

B Sampling from a four-layer neural network distribution

Appendix B.1 discusses how to draw from a four-layer neural network distribution using Gibbs sampling. Appendix B.2 shows another way to draw from a four-layer neural network: auxiliary variable Gibbs sampling.

B.1 Gibbs sampling from a four-layer neural network distribution

Suppose a density kernel of $X \in \mathbb{R}^n$ is given by

$$p(x) = \begin{cases} nn(x) & \text{if } x_i \in [\underline{x}_i, \bar{x}_i] \ \ \forall i = 1, \ldots, n \\ 0 & \text{else} \end{cases} \tag{70}$$

where $[\underline{x}_i, \bar{x}_i]$ is the interval to which X_i $(i = 1, \ldots, n)$ is restricted. Suppose the function $nn(x)$ corresponds to the following four-layer feed-forward neural network with n inputs x_i $(i = 1, \ldots, n)$, and H hidden cells:

$$nn(x) = \exp\left(\sum_{h=1}^{H} c_h \, plin\left(\sum_{i=1}^{n} a_{hi} x_i + b_h\right) + d\right), \tag{71}$$

where $plin : \mathbb{R} \to \mathbb{R}$ is the following piecewise-linear function:

$$plin(x) = \begin{cases} 0 & x < -1/2 \\ x + 1/2 & -1/2 \leq x \leq 1/2 \\ 1 & x > 1/2 \end{cases} \tag{72}$$

We rewrite the neural network density $nn(x) = nn(x_j, x_{-j})$ as

$$\begin{aligned} nn(x_j, x_{-j}) &= \exp\left(\sum_{h=1}^{H} c_h \, plin\left(\sum_{i=1}^{n} a_{hi} x_i + b_h\right) + d\right) \\ &\propto \exp\left(\sum_{h=1}^{H} c_h \, plin\left(a_{hj} x_j + \sum_{i=1, i \neq j}^{n} a_{hi} x_i + b_h\right)\right), \end{aligned}$$

which is a kernel of the conditional density of x_j given x_{-j}. Each hidden cell h $(h = 1, \ldots, H)$ has two points x_j, where its input $a_h' x + b_h$ moves from one of the intervals $(-\infty, -1/2)$, $[-1/2, 1/2]$ and $(1/2, \infty)$ to another one:

$$\begin{aligned} a_{hj} x_j + \sum_{i=1, i \neq j}^{n} a_{hi} x_i + b_h &= \pm\frac{1}{2} \Leftrightarrow \\ x_j &= \frac{1}{a_{hj}} \left(\pm\frac{1}{2} - \sum_{i=1, i \neq j}^{n} a_{hi} x_i - b_h\right) \end{aligned} \tag{73}$$

Consider only those 'changing points' $\tilde{x}_{j,k}$ ($k = 1, \ldots, m$ with $m \leq 2H$) that are in the interval of interest $[\underline{x}_j, \bar{x}_j]$, and order these m points such that:

$$\tilde{x}_{j,1} < \tilde{x}_{j,2} < \cdots < \tilde{x}_{j,m-1} < \tilde{x}_{j,m}$$

If we define $\tilde{x}_{j,0} = \underline{x}_j$ and $\tilde{x}_{j,m+1} = \bar{x}_j$, we have $m+1$ intervals $[\tilde{x}_{j,k}, \tilde{x}_{j,k+1}]$ ($k = 0, 1, \ldots, m$) on which a kernel of the conditional density of X_j given X_{-j} is given by:

$$nn(x_j, x_{-j}) \propto \exp(\tilde{a}_k x_j + \tilde{b}_k) \tag{74}$$

with

$$\begin{aligned} \tilde{a}_k &= \sum_{h=1}^{H} c_h a_{hj} D_{1,k,h} \\ \tilde{b}_k &= \sum_{h=1}^{H} \left\{ c_h \left(\sum_{i=1, i \neq j}^{n} a_{hi} x_i + b_h + \frac{1}{2} \right) D_{1,k,h} + c_h D_{2,k,h} \right\} \end{aligned}$$

where the dummies $D_{1,k,h}$ and $D_{2,k,h}$, defined by:

$$D_{1,k,h} = \begin{cases} 1 & \text{if } -\frac{1}{2} < \sum_{i=1}^{n} a_{hi} x_i + b_h < \frac{1}{2} \\ 0 & \text{else} \end{cases}$$

and

$$D_{2,k,h} = \begin{cases} 1 & \text{if } \sum_{i=1}^{n} a_{hi} x_i + b_h > \frac{1}{2} \\ 0 & \text{else} \end{cases}$$

are constant within each interval $[\tilde{x}_{j,k}, \tilde{x}_{j,k+1}]$. Equation (74) follows from the fact that:

$$\begin{aligned} & c_h \, plin \left(\sum_{i=1}^{n} a_{hi} x_i + b_h \right) = \\ = & \begin{cases} c_h & \text{if } D_{2,k,h} = 1 \\ (c_h a_{hj}) x_j + c_h \left(\sum_{i=1, i \neq j}^{n} a_{hi} x_i + b_h + \frac{1}{2} \right) & \text{if } D_{1,k,h} = 1 \\ 0 & \text{else} \end{cases} \end{aligned} \tag{75}$$

The primitive of (74) is given by

$$\int \exp(\tilde{a}_k x_j + \tilde{b}_k) dx_j = \begin{cases} \frac{1}{\tilde{a}_k} \exp(\tilde{a}_k x_j + \tilde{b}_k) + C_k & \text{if } \tilde{a}_k \neq 0 \\ \exp(\tilde{b}_k) x_j + C_k & \text{if } \tilde{a}_k = 0. \end{cases}$$

where C_k ($k = 0, 1, \ldots, m$) are integration constants. In order to obtain a proper kernel of the conditional CDF that starts at the value 0 and is continuous in the points $\tilde{x}_{j,k}$ ($k = 1, 2, \ldots, m$), we recursively determine the constants C_k in the following way:

$$C_0 = \begin{cases} -\frac{1}{\tilde{a}_0} \exp(\tilde{a}_0 \underline{x}_j + \tilde{b}_0) & \text{if } \tilde{a}_0 \neq 0 \\ -\exp(\tilde{b}_0) \, \underline{x}_j & \text{if } \tilde{a}_0 = 0. \end{cases}$$

and for $k = 1, 2, \ldots, m$:

$$C_k = C_{k-1} + \frac{1}{\tilde{a}_{k-1}} \exp(\tilde{a}_{k-1}\tilde{x}_{j,k} + \tilde{b}_{k-1}) - \frac{1}{\tilde{a}_k} \exp(\tilde{a}_k \tilde{x}_{j,k} + \tilde{b}_k)$$

if $\tilde{a}_{k-1}, \tilde{a}_k \neq 0$, and otherwise analogously.

After a kernel of the conditional CDF has been obtained, it is easy to sample X_j from its conditional distribution using the inverse transformation method. First, the 'scaling constant' S of the kernel is computed:

$$\begin{aligned} S &\equiv \int_{\underline{x}_j}^{\bar{x}_j} \exp\left(\sum_{h=1}^{H} c_h\, g\left(a_{hj}x_j + \sum_{i=1, i\neq j}^{n} a_{hi}x_i + b_h\right)\right) dx_j \\ &= \left\{ \begin{array}{ll} \frac{1}{\tilde{a}_m} \exp(\tilde{a}_m \bar{x}_j + \tilde{b}_m) + C_m & \text{if } \tilde{a}_m \neq 0 \\ \\ \exp(\tilde{b}_m)\bar{x}_j + C_m & \text{if } \tilde{a}_m = 0. \end{array} \right. \end{aligned} \tag{76}$$

Then we draw X_j by drawing $U \sim U(0,1)$, and solving

$$U = \frac{1}{S}\left(\frac{1}{\tilde{a}_k} \exp(\tilde{a}_k X_j + \tilde{b}_k) + C_k\right) \Leftrightarrow X_j = \frac{\log\left[\tilde{a}_k\left(S\,U - C_k\right)\right] - \tilde{b}_k}{\tilde{a}_k}$$

or

$$U = \frac{1}{S}\left(\exp(\tilde{b}_k)X_j + C_k\right) \Leftrightarrow X_j = \frac{S\,U - C_k}{\exp(\tilde{b}_k)}$$

depending on whether X_j falls in a region with $\tilde{a}_k = 0$ or not.

Since it is easy to draw X_j conditional on X_{-j} ($j = 1, \ldots, n$), it is easy to perform Gibbs sampling from a four-layer neural network distribution.

B.2 Auxiliary variable Gibbs sampling from a four-layer neural network distribution

Suppose a density kernel of $X \in \mathbb{R}^n$ is given by

$$p(x) = \left\{ \begin{array}{ll} nn(x) & \text{if } x_i \in [\underline{x}_i, \bar{x}_i] \ \ \forall i = 1, \ldots, n \\ 0 & \text{else} \end{array} \right. \tag{77}$$

where $[\underline{x}_i, \bar{x}_i]$ is the interval to which X_i ($i = 1, \ldots, n$) is restricted. Suppose the function $nn(x)$ corresponds to the following four-layer feed-forward neural network with n inputs x_i ($i = 1, \ldots, n$), and H hidden cells:

$$nn(x) = \exp\left(\sum_{h=1}^{H} c_h\, g\left(\sum_{i=1}^{n} a_{hi}x_i + b_h\right) + d\right), \tag{78}$$

where $g : \mathbb{R} \to \mathbb{R}$ is a monotonically increasing function taking its values in [0,1], which is invertible on the interval $(\underline{x}, \bar{x})$ where it takes its values in (0,1). We will denote this invertible function by $\tilde{g} : (\underline{x}, \bar{x}) \to (0,1)$ with inverse $\tilde{g}^{-1} : (0,1) \to (\underline{x}, \bar{x})$. Note that the interval $(\underline{x}, \bar{x})$ may be equal to $(-\infty, \infty)$.

Auxiliary variable Gibbs sampling is possible if the density kernel p can be decomposed as follows:

$$p(x) \propto \pi(x) \prod_{k=1}^{K} l_k(x), \tag{79}$$

where π is a density kernel from which sampling is easy, and l_k $(k = 1, \ldots, K)$ are non-negative functions of $x \in \mathbb{R}^n$. The trick is that a set $U = (U_1, \ldots, U_K)$ of auxiliary variables is introduced such that a kernel of the joint density of X and U is given by:

$$p(x, u) \propto \pi(x) \prod_{k=1}^{K} I\left\{0 < u_k < l_k(x)\right\}. \tag{80}$$

It is easily seen that (79) is a marginal density kernel corresponding to the joint density (80). Therefore one can sample $X \sim p(x)$ by sampling both X and U from (80) and forgetting U.

Kernels from the conditional distributions of X and U are easily obtained from the joint density kernel:

$$p(x|u) \propto \pi(x) I\left\{l_k(x) > u_k, k = 1, \ldots, K\right\} \tag{81}$$

$$p(u|x) \propto \prod_{k=1}^{K} I\left\{0 < u_k < l_k(x)\right\} \tag{82}$$

It follows from (81) and (82) that an iteration of the auxiliary variable Gibbs sampler consists of drawing X from a truncated version of an ‘easy’ distribution with density kernel π, and sampling U_k $(k = 1, \ldots, K)$ from K independent uniform distributions.

We rewrite (77) as:

$$p(x) \propto \prod_{i=1}^{n} I\left\{\underline{x}_i < x_i < \bar{x}_i\right\} \prod_{h=1}^{H} \exp\left(c_h\, g\left(\sum_{i=1}^{n} a_{hi} x_i + b_h\right)\right). \tag{83}$$

which has the shape of (79) with

$$\pi(x) = \prod_{i=1}^{n} I\left\{\underline{x}_i < x_i < \bar{x}_i\right\}, \tag{84}$$

$$l_h(x) = \exp\left(c_h\, g\left(\sum_{i=1}^{n} a_{hi} x_i + b_h\right)\right) \quad \text{for } h = 1, \ldots, H. \tag{85}$$

where $\pi(x)$ is the ‘easy’ density kernel of n independent variables X_i $(i = 1, \ldots, n)$ with distribution $U(\underline{x}_i, \bar{x}_i)$.

Drawing U conditionally on the values of X is straightforward. Combining (82) and (85), it follows that the elements U_h $(h = 1, \ldots, H)$ are drawn independently from the distributions:

$$U_h | X = x \;\sim\; U\left(0, \exp\left[c_h\, g\left(\sum_{i=1}^{n} a_{hi} x_i + b_h\right)\right]\right) \tag{86}$$

Drawing X conditionally on the values of U is a little harder. There are two approaches to do this. The first approach is to draw random vectors X from the 'unrestricted' distribution in (18), that is, drawing the elements X_i $(i = 1, \ldots, n)$ independently from $U(\underline{x}_i, \bar{x}_i)$, until the inequalities $l_h(x) > u_h$ $(h = 1, \ldots, H)$ are all satisfied. However, this acceptance-rejection approach is probably very inefficient. The second approach is to break up X, and sample the elements X_i $(i = 1, \ldots, n)$ conditionally on the values of U and the set of all other elements X_{-i}.

Combining (81), (84) and (85), we derive a density kernel of the conditional distribution of X_i given X_{-i} and U:

$$p(x_i|u, x_{-i}) \propto I\left\{\underline{x}_i < x_i < \bar{x}_i\right\} I\left\{l_h(x_i, x_{-i}) > u_h, h = 1, \ldots, H\right\} \tag{87}$$

We now take a closer look at the inequalities $l_h(x_i, x_{-i}) > u_h$ $(h = 1, \ldots, H)$. First, we can rule out that $c_h = 0$ for any h, since in this case we just delete the involved hidden cell.

Now suppose that $c_h > 0$. Then l_h takes its values in the interval $[1, \exp(c_h)]$, as g takes its values in [0,1]. This means that u_h has to lie in $(0, \exp(c_h))$, as we have $0 < u_h < l_h \leq \exp(c_h)$. We conclude that if $u_h \in [0, 1]$, the inequality $l_h(x_i, x_{-i}) > u_h$ is always satisfied, that is, it does not imply any restriction on the value of x_i. However, if $u_h \in (1, \exp(c_h))$, the inequality can be rewritten as:

$$\begin{aligned} \exp\left(c_h\, g\left(\sum_{j=1}^{n} a_{hj}x_j + b_h\right)\right) &> u_h \Leftrightarrow \\ g\left(\sum_{j=1}^{n} a_{hj}x_j + b_h\right) &> \frac{\log(u_h)}{c_h}. \end{aligned}$$

Since $u_h \in (1, \exp(c_h))$ implies that $\log(u_h)/c_h \in (0, 1)$, we rewrite this as:

$$\sum_{j=1}^{n} a_{hj}x_j + b_h > \tilde{g}^{-1}\left(\frac{\log(u_h)}{c_h}\right),$$

which is equivalent with

$$a_{hi}x_i > \tilde{g}^{-1}\left(\frac{\log(u_h)}{c_h}\right) - \left(\sum_{j=1, j\neq i}^{n} a_{hj}x_j + b_h\right),$$

so that if $a_{hi} > 0$, we obtain a lower bound:

$$x_i > \frac{1}{a_{hi}}\left(\tilde{g}^{-1}\left(\frac{\log(u_h)}{c_h}\right) - \left(\sum_{j=1, j\neq i}^{n} a_{hj}x_j + b_h\right)\right), \tag{88}$$

whereas if $a_{hi} < 0$, we get an upper bound:

$$x_i < \frac{1}{a_{hi}}\left(\tilde{g}^{-1}\left(\frac{\log(u_h)}{c_h}\right) - \left(\sum_{j=1, j\neq i}^{n} a_{hj}x_j + b_h\right)\right). \tag{89}$$

Now suppose that $c_h < 0$. Then l_h takes its values in the interval $[\exp(c_h), 1]$, so that u_h lies in $(0, 1)$. We conclude that if $u_h \in [0, \exp(c_h)]$, the inequality $l_h(x_i, x_{-i}) > u_h$ does not imply any restriction on the value of x_i. However, if $u_h \in (\exp(c_h), 1)$, the inequality can be rewritten as:

$$\begin{aligned} \exp\left(c_h\, g\left(\sum_{j=1}^{n} a_{hj}x_j + b_h\right)\right) &> u_h \Leftrightarrow \\ a_{hi}x_i &< \tilde{g}^{-1}\left(\frac{\log(u_h)}{c_h}\right) - \left(\sum_{j=1, j\neq i}^{n} a_{hj}x_j + b_h\right), \end{aligned}$$

so that if $a_{hi} > 0$, we obtain an upper bound:

$$x_i < \frac{1}{a_{hi}}\left(\tilde{g}^{-1}\left(\frac{\log(u_h)}{c_h}\right) - \left(\sum_{j=1, j\neq i}^{n} a_{hj}x_j + b_h\right)\right), \tag{90}$$

whereas if $a_{hi} < 0$, we get a lower bound:

$$x_i > \frac{1}{a_{hi}}\left(\tilde{g}^{-1}\left(\frac{\log(u_h)}{c_h}\right) - \left(\sum_{j=1, j\neq i}^{n} a_{hj}x_j + b_h\right)\right). \tag{91}$$

Note that the conditions $u_h \in (1, \exp(c_h))$ for $c_h > 0$ and $u_h \in (\exp(c_h), 1)$ for $c_h < 0$ can be summarized by the condition

$$\frac{\log(u_h)}{c_h} \in (0, 1),$$

the interval for which the inverse $\tilde{g}^{-1}$ exists. So, it follows from (88), (89), (90) and (91) that the conditions

$$l_h(x_i, x_{-i}) > u_h,\ \ h = 1, \ldots, H$$

in (87) are equivalent with

$$x_i > \frac{1}{a_{hi}}\left(\tilde{g}^{-1}\left(\frac{\log(u_h)}{c_h}\right) - \left(\sum_{j=1, j\neq i}^{n} a_{hj}x_j + b_h\right)\right),$$

for those h with $c_h a_{hi} > 0$ and $\log(u_h)/c_h \in (0, 1)$, and

$$x_i < \frac{1}{a_{hi}}\left(\tilde{g}^{-1}\left(\frac{\log(u_h)}{c_h}\right) - \left(\sum_{j=1, j\neq i}^{n} a_{hj}x_j + b_h\right)\right),$$

for those h with $c_h a_{hi} < 0$ and $\log(u_h)/c_h \in (0, 1)$.

Note that if we consider $l_h(x_i, x_{-i})$ as a function of x_i for given values of x_{-i}, denoted by $l_{h,x_{-i}}(x_i)$, then the inverse $l^{-1}_{h,x_{-i}}$ (if it exists) is given by:

$$l^{-1}_{h,x_{-i}}(u_h) = \frac{1}{a_{hi}}\left(\tilde{g}^{-1}\left(\frac{\log(u_h)}{c_h}\right) - \left(\sum_{j=1, j\neq i}^{n} a_{hj}x_j + b_h\right)\right). \tag{92}$$

We conclude that (87) is a density kernel of the distribution

$$X_i|U = u, X_{-i} = x_{-i} \ \sim\ U(x_{i,LB}(u, x_{-i}), x_{i,UB}(u, x_{-i})), \tag{93}$$

with

$$x_{i,LB}(u, x_{-i}) = \max\left\{ \max_{1\leq h\leq H} \left\{ l^{-1}_{h,x_{-i}}(u_h) \middle| c_h a_{hi} > 0, \frac{\log(u_h)}{c_h} \in (0,1) \right\}, \underline{x}_i \right\}$$

$$x_{i,UB}(u, x_{-i}) = \min\left\{ \min_{1\leq h\leq H} \left\{ l^{-1}_{h,x_{-i}}(u_h) \middle| c_h a_{hi} < 0, \frac{\log(u_h)}{c_h} \in (0,1) \right\}, \bar{x}_i \right\},$$

where $l^{-1}_{h,x_{-i}}(u_h)$ is given by (92), and where $[\underline{x}_i, \bar{x}_i]$ is the interval to which X_i $(i = 1, \dots, n)$ is a priori restricted.

The auxiliary variable Gibbs sampling procedure is now given by:

```
Initialization: Choose feasible $x^0 = (x^0_1, \dots, x^0_n)$.

Do for $j = 1, 2, \dots, m$

    Do for $h = 1, 2, \dots, H$
        Obtain $u^j_h \sim U_h|X = x^{j-1}$ from (86).
    Do for $i = 1, 2, \dots, n$
        Obtain $x^j_i \sim X_i|U = u^j, X_{-i} = x^{j-1}_{-i}$ from (93).
```

Here x^{j-1}_{-i} denotes

$$x^{j-1}_{-i} = x^j_1, \dots, x^j_{i-1}, x^{j-1}_{i+1}, \dots, x^{j-1}_n,$$

the set of all components except x_i at their current values. Note that this procedure only requires drawing from uniform distributions, which is done easily and fast.

References

[1] Bauwens, L. and H.K. van Dijk (1989): "Bayesian limited information analysis revisited". In: J. J. Gabszewicz et al. (eds), *Economic Decision-Making: Games, Econometrics and Optimisation*, North-Holland, Amsterdam.

[2] Bauwens, L., M. Lubrano and J.-F. Richard (1999): *Bayesian Inference in Dynamic Econometric Models*, Oxford University Press.

[3] Bauwens, L., C.S. Bos, H.K. van Dijk and R.D. van Oest (2002): "Adaptive Polar Sampling: A class of flexible and robust Monte Carlo integration methods", Econometric Institute report 2002-27, Erasmus University Rotterdam.

[4] Chib, S. and E. Greenberg (1996): "Markov Chain Monte Carlo Simulation Methods in Econometrics", *Econometric Theory*, 12(3), 409-431.

[5] Gallant, A.R. and H. White (1989): "There exists a neural network that does not make avoidable mistakes", in *Proc. of the International Conference on Neural Networks*, San Diego, 1988 (IEEE Press, New York).

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

[7] Geweke, J. (1999): "Using Simulation Methods for Bayesian Econometric Models: Inference, Development, and Communication", *Econometric Reviews*, 18(1), 1-73.

[8] Gilks, W. and G. Roberts (1996): "Strategies for Improving MCMC", in *Markov Chain Monte Carlo in Practice* ed. by R.S. Gilks, W.R., and D. Spiegelhalter. Chapman and Hall/CRC.

[9] Hammersley, J. and D. Handscomb (1964): "Monte Carlo Methods". Chapman and Hall, London.

[10] Hastings, W.K. (1970): "Monte Carlo Sampling Methods using Markov Chains and their Applications", *Biometrika*, 57, 97-109.

[11] Hecht-Nielsen, R. (1987): "Kolmogorov mapping neural network existence theorem", in *Proc. IEEE First International Conference on Neural Networks*, San Diego, 1987, 11-13.

[12] Hamilton, J.D. (1989): "A New Approach to the Econometric Analysis of Nonstationary Time Series and Business Cycles", *Econometrica*, 57, 357-384.

[13] Hobert, J.P. and G. Casella (1996): "The Effect of Improper Priors on Gibbs Sampling in Hierarchical Linear Mixed Models", *Journal of the American Statistical Association*, 91(436), 1461-1473.

[14] Hornik, K., M. Stinchcombe, and H. White (1989): "Multilayer feedforward networks are universal approximators", *Neural Networks*, Vol. 2, 359-366.

[15] Kleibergen, F.R., and H.K. Van Dijk (1994): "On the Shape of the Likelihood/Posterior in Cointegration Models", *Econometric Theory*, 10(3-4), 514-551.

[16] Kleibergen, F.R., and H.K. Van Dijk (1998): "Bayesian Simultaneous Equations Analysis using Reduced Rank Structures", *Econometric Theory*, 14(6), 701-743.

[17] Kloek, T., and H.K. Van Dijk (1978): "Bayesian estimates of equation system parameters: an application of integration by Monte Carlo", *Econometrica*, 46, 1-19.

[18] Kolmogorov, A.N. (1957): "On the representation of continuous functions of many variables by superposition of continuous functions of one variable and addition", *American Mathematical Monthly Translation*, Vol. 28, pp 55-59. (Russian original in Doklady Akademii Nauk SSSR, 144, 953-956)

[19] Leshno, M., Lin, V.Y., Pinkus, A. and Schocken, S. (1993): "Multilayer Feedforward Networks With a Nonpolynomial Activation Function Can Approximate Any Function", *Neural networks*, Vol. 6, 861-867.

[20] Liesenfeld, R. and J.-F. Richard (2002): "Univariate and Multivariate Stochastic Volatility Models: Estimation and Diagnostics", Discussion paper, University of Tubingen.

[21] Metropolis, N., A.W. Rosenbluth, M.N. Rosenbluth, A.H. Teller, and E. Teller (1953): "Equations of State Calculations by Fast Computing Machines", *Journal of Chemical Physics*, 21, 1087-1091.

[22] Paap, R. and H.K. van Dijk (2002): "Bayes Estimates of Markov Trends in Possibly Cointegrated Series: An Application to US Consumption and Income", Econometric Institute report 2002-42, Erasmus University Rotterdam.

[23] Pesaran, M.H. and R. Smith (1995): "Estimation of Long-Run Relationships from Dynamic Heterogeneous Panels", *Journal of Econometrics*, 68, 79-113.

[24] Richard, J.-F. (2002): "Efficient High-dimensional Monte Carlo Importance Sampling", Discussion paper, University of Pittsburgh.

[25] Schotman, P.C. and H.K. van Dijk (1991): "A Bayesian Analysis of the Unit Root in Real Exchange Rates", *Journal of Econometrics*, 49, 195-238.

[26] Shephard, N. (1996): "Statistical aspects of ARCH and stochastic volatility", in *Time Series Models with Econometric, Finance and Other Applications*, ed. by D.R. Cox, D.V. Hinkley and O.E. Barndorff-Nielson, Chapman and Hall, London.

[27] Tanner, M.A. and W.H. Wong (1987): "The Calculation of Posterior Distributions by Data Augmentation" (with discussion), *Journal of the American Statistical Association*, 82, 528-550.

[28] Tierney, L. (1994): "Markov Chains for Exploring Posterior Distributions", *Annals of Statistics*, 22, 1701-1762.

[29] Van Dijk, H.K., and T. Kloek (1980): “Further experience in Bayesian analysis using Monte Carlo integration”, *Journal of Econometrics*, 14, 307-328.

[30] Van Dijk, H.K., and T. Kloek (1984): “Experiments with some alternatives for simple importance sampling in Monte Carlo integration”, in *Bayesian Statistics 2*, ed. by J. M. Bernardo, M. Degroot, D. Lindley, and A. F. M. Smith, Amsterdam, North-Holland.

[31] Zellner, A. (1971): *An introduction to Bayesian inference in econometrics.* Wiley, New York.