Estimating Uncertainty of Streamflow Simulation using Bayesian Neural Networks

Xuesong Zhang[1], Faming Liang[2], Raghavan Srinivasan[1], and Michael Van Liew[3]

[1]Spatial Sciences Laboratory, Department of Ecosystem Sciences and Management, Texas A&M University, College Station, TX 77843, USA

[2]Department of Statistics, Texas A&M University, College Station, TX 77843, USA

[3]Montana Dept Environmental Quality, Helena, MT 59620, USA

Corresponding author:

Raghavan Srinivasan, Spatial Sciences Laboratory, 1500 Research Parkway, Suite B223, 2120 TAMU, College Station, TX 77843-2120

Phone: (979) 845-5069

Fax: (979) 862-2607

Email: r-srinivasan@tamu.edu

Abstract: Recent studies have shown that Bayesian Neural Networks (BNNs) are powerful tools for providing reliable hydrologic prediction and quantifying the prediction uncertainty. The reasonable estimation of the prediction uncertainty, a valuable for decision making to address water resources management and design problems, is influenced by the techniques used to deal with different uncertainty sources. In this study, four types of BNNs with different treatments of the uncertainties related to parameters (neural network's weights) and model structures were applied for uncertainty estimation of streamflow simulation in two USDA ARS watersheds (Little River Experimental Watershed in GA and Reynolds Creek Experimental Watershed in ID). An advanced Markov Chain Monte Carlo (MCMC) algorithm - Evolutionary Monte Carlo (EMC) was used to train the BNNs and estimate uncertainty limits of streamflow simulation. The results obtained in these two case study watersheds show that the 95% uncertainty limits estimated by different types of BNNs are different from each other. The BNNs that only consider the parameter uncertainty with non-informative prior knowledge contain the least number of observed streamflow data in their 95% uncertainty bound. By considering variable model structure and informative prior knowledge, the BNNs can provide more reasonable quantification of the uncertainty of streamflow simulation. This study stresses the need for improving understanding and quantifying methods of different uncertainty sources for effective estimation of uncertainty of hydrologic simulation using BNNs.

Keywords: Artificial Neural Networks, Bayesian Model Averaging, Bayesian Neural Networks, Evolutionary Monte Carlo, Hydrologic Simulation, Streamflow, Uncertainty

1 Introduction

With the powerful capacity of capturing non-linear relationships between inputs and outputs data without requiring an in-depth understanding of the underlying physical processes [Kingston *et al.*, 2005], artificial neural networks (ANNs) have been successfully applied in a wide range of hydrologic problems [ASCE task Committee on Application of artificial Neural Networks in Hydrology, 2000a, 2000b]. For example, ANNs have been used for predicting flash flood and attendant water qualities [Sahoo *et al.*, 2006], runoff and sediment-yield modeling [Raghuwanshi *et al.*, 2006], evaporation estimation [Keskin and Terzi, 2006], and ice growth [Seidou *et al.*, 2006]. Many researchers have also modified ANNs to improve their ability to more accurately model hydrologic variables of interest [e.g. Jain and Srinivasulu, 2004; Chetan and Sudheer, 2006; Parasuraman *et al.*, 2006; Nayak *et al.*, 2007]. Although ANNs have been widely used in hydrologic modeling, one major limitation of ANNs is that the neural networks are trained by maximizing a likelihood function of the parameters and hence the uncertainty of the predicted variables are seldom quantified [Maier and Dandy, 2000; Dawson and Wilby, 2001; Coulibaly *et al.*, 2001; Kingston *et al.*, 2005; and Khan and Coulibaly, 2006]. Not considering uncertainty of model parameters or the uncertainty about the relationship between input and output simulated by the networks leads to the failure to evaluate the predictive uncertainty and limits the usability of ANNs in real-world hydrologic problem [Kingston *et al.*, 2005; Khan and Coulibaly, 2006].

Bayesian analysis of the neural networks can yield predictive distribution of the variables of interest and make the computation of confident intervals possible [Lampinen and Vehtari, 2001]. Since the Bayesian evidence framework proposed by MackKay

[1992], Bayesian Neural Networks (BNNs) have been widely applied in training, model selection, and prediction [e.g., Neal, 1996; Bishop, 1995; Müller and Rios Insua, 1998; Holmes and Mallick, 1998; de Freitas *et al*., 2000; Liang, 2003; Liang and Kuk, 2004; Liang, 2005a]. Recently, the Bayesian methodology for quantifying predictive uncertainty of ANNs has been extended and applied for hydrologic modeling. For example, Kingston *et al.* [2005] attributed the prediction uncertainty of ANNs to the uncertainty in the weights (the connections and biases) of the neural networks. They combined the traditional ANNs with the adaptive Metropolis algorithm [Haario *et al*., 2001] to sample a large number of sets of neural network weights observing the posterior distributions, which were used to determine predictive limits and calculate mean prediction. Khan and Coulibaly [2006] defined the posterior distribution of network weights through a Gaussian prior distribution and a Gaussian noise model, and obtained the predictive distribution of the network outputs by integrating over the posterior distribution with the assumption that posterior of network weights is approximated to Gaussian during prediction. In the above two case studies by Kingston et al. (2005) and Khan and Coulibaly [2006], it was shown that the BNNs outperformed the traditional deterministic ANNs in terms of prediction accuracy and that the BNNs can also estimate the predictive confidence intervals that indicate the quality of the prediction.

The reasonable estimates of predictive uncertainty of hydrologic prediction are valuable to water resources and other relevant decision making processes [Liu and Gupta, 2007]. Usually, water management projects are planned and designed using scenarios that fall at the conservative end of the range of plausible outcomes. Over estimation of uncertainty can result in over design of mitigation measures, while under estimation of

uncertainty can lead to inadequate preparation for potential situations. There is a variety of uncertainty analysis methods differing in philosophy, assumptions, and sampling strategies, and the understanding and quantification of different uncertainty sources can influence the estimation of the predictive uncertainty of hydrologic modeling [e.g. Beven and Binley, 1992; Beven and Freer, 2001; Wagner and Gupta, 2005; Beven, 2006; Vrugt and Robinson, 2007; Kavetski *et al.*, 2006; Ajami *et al.*, 2007]. Knowledge of the effect of the differences between various methods is still inadequate [Wagner and Gupta, 2005]. As a relatively new technique in hydrologic modeling, BNNs have not been widely applied to uncertainty estimation of hydrologic simulations. In the previous applications of BNNs in hydrologic modeling, the neural networks' weights were usually taken as the major source of uncertainty of neural networks prediction. But the structural error inherent in any model cannot be avoided in ensemble strategies using multi-parameter sets [Georgakakos *et al.*, 2004; Ajami *et al.*, 2007; Duan *et al.*, 2007; Vrugt *et al.*, 2007]. Many studies in hydrologic modeling have shown that model structural error can be one significant component of the overall predictive uncertainty [Wagner and Gupta 2005]. Understanding or the prior knowledge about the error characteristics that describe the probability distribution of the different uncertainty sources is important for effective quantification of the predictive uncertainty [Liu and Gupta, 2007]. In the framework outlined by several studies [e.g. Wagner and Gupta, 2005; Pappenberger and Beven, 2007; Liu and Gupta, 2007] for uncertainty quantification of hydrologic modeling, the understanding or prior knowledge of the uncertainty sources were emphasized.

In this study, the major objective was to evaluate the effect of different treatments of uncertainties related to parameters (networks' weights) and structures on the uncertainty

estimation of streamflow simulation using BNNs. To consider the uncertainties associated with neural networks' structures, the neural networks' connections were allowed to be variable. Non-informative and informative prior knowledge of neural networks' weights and structures was also obtained based on previous studies and incorporated into the BNNs. The different BNNs (with variable or fixed model structure, informative or non-informative prior knowledge) were applied in two case study United States Department of Agriculture, Agricultural Research Service (USDA ARS) experimental watersheds (Little River, GA and Reynolds Creek, ID) for daily streamflow simulation to derive results for analysis and discussion. The remainder of this paper is organized as follows. Section two provides a brief description of the BNNs (including neural networks' structure, the methods used to quantify the uncertainties associated with neural network weights and structures, an advanced Markov Chain Monte Carlo (MCMC) method - Evolutionary Monte Carlo (EMC)), and study area characteristics. Section three presents and discusses the application of BNNs with different considerations of uncertainties related to parameters and structures for streamflow simulation in the two case study watersheds. The generalization ability of BNNs is compared with deterministic ANNs, and the 95% uncertainty limits estimated by different BNNs are compared and discussed. Finally, a summary with conclusions is provided in section four.

2. Methods and Materials

2.1 Bayesian Neural Networks (BNNs)

2.1.1 Neural Networks Structure

Neural networks are universal approximates that have been widely used to simulate complex and nonlinear relationships between input and output data. The input data vector $\mathbf{x}_t$ is mapped to the target variable y_t in the form of $y_t = f(\mathbf{x}_t) + \varepsilon$, where $f(\mathbf{x}_t)$ is neural network function, ε is random noise term with zero mean and constant variance σ^2. Figure 1 shows a fully connected three layer feed-forward neural network with four inputs, four hidden units and one output. This network can be used to approximate the variable of interest using a function with the form

$$f(\mathbf{x}_t) = \alpha_0 + \sum_{i=1}^{p} x_{it}\alpha_i + \sum_{j}^{M} \beta_j \psi(\gamma_{j0} + \sum_{i=1}^{p} x_{it}\gamma_{ji}) \qquad (1)$$

where $\mathbf{x}_t$ is the input data vector at time *t*, p is the dimension of $\mathbf{x}_t$, x_{it} is the *i*th component of $\mathbf{x}_t$, M is the number of hidden units, α_0 denotes the bias of the output unit, α_i denotes the weight that directly connects the *i*th input unit to the output unit, β_j is the weight that connects the *j*th hidden unit to the output unit, γ_{j0} is the bias of the *j*th hidden unit, γ_{ji} denotes the weight on the connection from the *i*th input to the *j*th hidden unit, and $\psi(\cdot)$ is the activation function of the hidden units. The biases and connections need to be optimized to infer an acceptable approximation of the relationship underlying a system that relates a set of input variables to the dependent variables of interest. Usually, the neural network structure is fixed, which means that the number of connections

between the neurons is fixed. A set of indication functions can be linked with each connection to represent the validity of a specific connection [Liang and Kuk, 2004]. Then, the above neural network model form can be transformed to:

$$f(\mathbf{x}_t) = \alpha_0 I_{\alpha_0} + \sum_{i=1}^{p} x_{it}\alpha_i I_{\alpha_i} + \sum_{j}^{M} \beta_j I_{\beta_j} \psi(\gamma_{j0} I_{\gamma_{j0}} + \sum_{i=1}^{p} x_{it}\gamma_{ji} I_{\gamma_{ji}}) \qquad (2)$$

where I_ζ is the indicator function associated with the connection ζ. If I_ζ=1, then the connection is in effect, otherwise, I_ζ =0 and the connection is not effective. The activation function ($\psi(\cdot)$) applied in this study is the hyperbolic tangent function. The $\tanh(\cdot)$ function ensures that the output of a hidden unit is 0 if all connections to the hidden unit from input units have been eliminated. Let $\mathbf{\Lambda}$ be the vector consisting of all indicators in equation (2), which specifies the structure of the network. Let $\alpha = (\alpha_0, \alpha_1, \ldots, \alpha_p)$, $\beta = (\beta_0, \beta_1, \ldots, \beta_M)$, $\gamma_j = (\gamma_{j0}, \gamma_{j1}, \ldots, \gamma_{jp})$, $\gamma = (\gamma_1, \gamma_2, \ldots, \gamma_M)$, and $\mathbf{\theta} = (\alpha_\Lambda, \beta_\Lambda, \gamma_\Lambda, \sigma^2)$, where α_Λ, β_Λ, and γ_Λ denote the non-zero subsets of α, β, and γ, respectively. Thus the combination of ($\mathbf{\theta}, \mathbf{\Lambda}$) completely defines equation (2). In the following, a neural network model is defined by ($\mathbf{\theta}, \mathbf{\Lambda}$), and $f(\mathbf{x}_t)$ is represented by $f(\mathbf{x}_t, \mathbf{\theta}, \mathbf{\Lambda})$. Equation (1) is a special case of equation (2) with all connection being effective. The major difference between equation (1) and equation (2) is that equation (2) is trained by sampling from the joint posterior of the neural network structures and weights while equation (1) is trained by sampling from the posterior of the weights.

2.1.2 Bayesian training of neural networks

In the traditional deterministic training of a neural network, a single set of optimal ($\mathbf{\theta}, \mathbf{\Lambda}$) is identified that is most likely to reproduce the observed target data. From the

Bayesian viewpoint, the training of neural networks can be taken as a problem of inference. The key principle of Bayesian approach is to construct the posterior probability distribution of $(\boldsymbol{\theta}, \boldsymbol{\Lambda})$ given the observed input and target data sets. In the Bayesian training framework, the observed data and prior knowledge of parameters and model structure were applied to derive the posterior distribution of models $(\boldsymbol{\theta}, \boldsymbol{\Lambda})$ for inference. Given the training data sets $D = \{(\mathbf{x}_1, \mathbf{y}_1), (\mathbf{x}_2, \mathbf{y}_2), \ldots, (\mathbf{x}_n, \mathbf{y}_n)\}$, the posterior distribution of the weights and model structure $(\boldsymbol{\theta}, \boldsymbol{\Lambda})$ is defined as:

$$p(\boldsymbol{\theta}, \boldsymbol{\Lambda} \mid D) = \frac{p(D \mid \boldsymbol{\theta}, \boldsymbol{\Lambda}) \pi(\boldsymbol{\theta}, \boldsymbol{\Lambda})}{\int p(D \mid \boldsymbol{\theta}, \boldsymbol{\Lambda}) \pi(\boldsymbol{\theta}, \boldsymbol{\Lambda}) d(\boldsymbol{\theta}, \boldsymbol{\Lambda})} \tag{3}$$

Where $p(\boldsymbol{\theta}, \boldsymbol{\Lambda} \mid D)$ is the posterior probability distribution of $(\boldsymbol{\theta}, \boldsymbol{\Lambda})$ given observed data D, $\pi(\boldsymbol{\theta}, \boldsymbol{\Lambda})$ is the prior probability distribution of $(\boldsymbol{\theta}, \boldsymbol{\Lambda})$, $\int p(D \mid \boldsymbol{\theta}, \boldsymbol{\Lambda}) \pi(\boldsymbol{\theta}, \boldsymbol{\Lambda}) d(\boldsymbol{\theta}, \boldsymbol{\Lambda})$ is the normalizing constant, and $p(D \mid \boldsymbol{\theta}, \boldsymbol{\Lambda})$ is the likelihood function of $(\boldsymbol{\theta}, \boldsymbol{\Lambda})$, which is denoted as $L(\boldsymbol{\theta}, \boldsymbol{\Lambda})$ in the following. Through integrating the predictions of the model with respect to the posterior distribution of the model $(\boldsymbol{\theta}, \boldsymbol{\Lambda})$, the posterior predictive distribution of output $\mathbf{y}_{new}$ for the new input $\mathbf{x}_{new}$, is [Lampinen and Vehtari, 2001],

$$p(\mathbf{y}_{new} \mid \mathbf{x}_{new}, D) = \int p(\mathbf{y}_{new} \mid \mathbf{x}_{new}, \boldsymbol{\theta}, \boldsymbol{\Lambda}) \, p(\boldsymbol{\theta}, \boldsymbol{\Lambda} \mid D) d(\boldsymbol{\theta}, \boldsymbol{\Lambda}) \tag{4}$$

The expectation of the posterior prediction distribution in equation (4) is

$$\hat{\mathbf{y}}_{new} = E(\mathbf{y}_{new} \mid \mathbf{x}_{new}, D) = \int f(\mathbf{x}_{new}, \boldsymbol{\theta}, \boldsymbol{\Lambda}) \, p(\boldsymbol{\theta}, \boldsymbol{\Lambda} \mid D) d(\boldsymbol{\theta}, \boldsymbol{\Lambda}) \tag{5}$$

One major challenge in the Bayesian analysis of neural networks is evaluating integrals for posterior distribution and predictive distribution of network outputs [Khan and Coulibaly, 2006]. Usually, the posterior distribution of weights and model structures of neural networks is very complex and multimodal [Neal, 1996; Kingston *et al.*, 2005;

Khan and Coulibaly, 2006; Liang, 2005a, 2005b], and it is difficult to sample from the complex posterior distribution and generate candidate weights and model structures. In this case, the MCMC methods are usually used as tools for sampling the posterior probability of model structures and parameters of BNNs. In MCMC, the complex integrals in the marginalization are approximated via drawing samples from the joint probability distribution of $(\boldsymbol{\theta}, \boldsymbol{\Lambda})$, and $\hat{\mathbf{y}}_{new}$ can be approximated using a sample of the $(\boldsymbol{\theta}, \boldsymbol{\Lambda})$ drawn from the posterior probability distribution of $(\boldsymbol{\theta}, \boldsymbol{\Lambda})$ [Lampinen and Vehtari, 2001],

$$\hat{\mathbf{y}}_{new} = \frac{1}{K}\sum_{i=1}^{K} f(\mathbf{x}_{new}, \boldsymbol{\theta}_i, \boldsymbol{\Lambda}_i) \tag{6}$$

where, K denotes the number of all models $(\boldsymbol{\theta}, \boldsymbol{\Lambda})$ under consideration.

2.1.3 Posterior probability distribution of neural networks

In order to conduct Bayesian analysis of a neural network, the prior probability distribution $\pi(\boldsymbol{\theta}, \boldsymbol{\Lambda})$ and likelihood function $L(\boldsymbol{\theta}, \boldsymbol{\Lambda})$ in equation (3) needs to be specified. A popular method to specify the likelihood function is to assume the model residuals are normally and independently distributed with zero mean and constant variance σ^2. This leads to the following likelihood function:

$$L(\boldsymbol{\theta}, \boldsymbol{\Lambda}) = p(\mathbf{y} \mid \boldsymbol{\theta}, \boldsymbol{\Lambda}, \mathbf{x}) = \left\{ \prod_{t=1}^{n} \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{1}{2\sigma^2}(y_t - f(\mathbf{x}_t))^2\right) \right\} \tag{7}$$

Taking logarithm, we have

$$\log(L(\boldsymbol{\theta}, \boldsymbol{\Lambda})) = \text{constant} - \frac{1}{2\sigma^2}\sum_{t=1}^{n}\left(y_t - f(\mathbf{x}_t, \boldsymbol{\theta}, \boldsymbol{\Lambda})\right)^2 - \frac{n}{2}\log(\sigma^2) \tag{8}$$

where n is the number of observed target data, and σ^2 is referred to as hyperparameter which is assumed to observe Inverse Gamma (IG) distribution ($\sigma^2 \sim IG(v_1, v_2)$) with v_1 and v_2 are the shape parameter and scale parameter, respectively. As far as the prior distribution of $(\boldsymbol{\theta}, \boldsymbol{\Lambda})$, a convenient way is to assume non-informative prior distribution to represent the vague prior knowledge, which allows the posterior distributions of $(\boldsymbol{\theta}, \boldsymbol{\Lambda})$ to be determined by the observed data [Neal, 1996; Lampinen and Vehtari, 2001; Kingston, 2005]. A wide uniform prior, symmetric around zero, was used as the non-informative prior. With the assumption of non-informative prior distribution, the posterior probability of $(\boldsymbol{\theta}, \boldsymbol{\Lambda})$ can be formulated as equation (7) with the log form of equation (8). This form of posterior probability function can be applied for both fixed and variable neural network structures.

To some extent, the quantification of uncertainty is dependent on understanding prior knowledge of uncertainty. In practical application of neural networks, incorporating human prior knowledge in neural networks models was suggested to improve their performance [Wang, 1995; Müler and Insua, 1998; Liang, 2005b]. Usually, the large weights and bias values and complex model structures are penalized. In this paper, we follow Liang [2005b] to assume the following prior distributions for the weights: $\alpha_i \sim N(0, {\sigma_\alpha}^2)$ for $i = 0, \ldots, p$, $\beta_j \sim N(0, {\sigma_\beta}^2)$ for $j = 0, \ldots, M$, $\gamma_{ij} \sim N(0, {\sigma_\gamma}^2)$ for $j = 0, \ldots, M$ and $i = 0, \ldots, p$, where ${\sigma_\alpha}^2$, ${\sigma_\beta}^2$, and ${\sigma_\gamma}^2$ are hyper-parameters to be specified by users. By assuming that the components of $\boldsymbol{\theta}$ are a priori independent, we have

$$\log P(\boldsymbol{\theta} \mid D, \boldsymbol{\Lambda}) = \text{Constant} - (\frac{n}{2} + \nu_1 + 1)\log \sigma^2 - \frac{\nu_2}{\sigma^2} - \frac{1}{2\sigma^2}\sum_{t=1}^{n}(y_t - f(\mathbf{x}_t))^2 - \frac{1}{2}\sum_{i=0}^{p} I_{\alpha i}(\log {\sigma_\alpha}^2 + \frac{{\alpha_i}^2}{{\sigma_\alpha}^2})$$

$$-\frac{1}{2}\sum_{j=1}^{M} I_{\beta j}\Phi(\sum_{i=0}^{p} I_{\gamma_{ji}})(\log {\sigma_\beta}^2 + \frac{{\beta_j}^2}{{\sigma_\beta}^2}) - \frac{1}{2}\sum_{j=1}^{M}\sum_{i=0}^{p} I_{\beta_j} I_{\gamma_{ji}}(\log {\sigma_\gamma}^2 + \frac{{\gamma_{ji}}^2}{{\sigma_\gamma}^2}) - \frac{m}{2}\log(2\pi) \tag{9}$$

Where $m = \sum_{i=0}^{p} I_{\alpha i} + \sum_{j=1}^{M} I_{\beta j}\Phi(\sum_{i=0}^{p} I_{\gamma_{ji}}) + \sum_{j=1}^{M}\sum_{i=0}^{p} I_{\beta_j} I_{\gamma_{ji}}$ is the total number of effective connections, $\Phi(t)$ is 1 if $t>0$ and 0 otherwise. For fixed model structure, all indicator functions are equal to 1 and m is a constant.

The prior knowledge of the structure of neural networks can also be taken into account. As in Müler and Insua [1998] and Liang [2005b], the neural network's structure $\boldsymbol{\Lambda}$ is set to be subject to a prior probability that is proportional to a truncated Poisson:

$$p(\Lambda) = \begin{cases} \frac{1}{Z}\frac{\lambda^m}{m!}, & m = 3,4,\ldots,U \\ 0, & \text{otherwise} \end{cases} \tag{10}$$

where λ is a hyper-parameter, $U = (M+1)(p+1) + M$ is the number of connections of the full model in which all connections are valid, $Z = \sum_{\Lambda \in \Omega} \lambda^m / m!$, and Ω is a set of all possible model structures with $m = 3,4,\ldots,U$. The minimum number of m is set to be three to limit the network size. Furthermore, we assume that the prior distributions of $\boldsymbol{\theta}$ and $\boldsymbol{\Lambda}$ are independent. Then, the posterior distribution of $(\boldsymbol{\theta}, \boldsymbol{\Lambda})$ can be formalized by multiplying the prior distributions of $\boldsymbol{\theta}$ and $\boldsymbol{\Lambda}$ and $L(\boldsymbol{\theta}, \boldsymbol{\Lambda})$. The log form of this posterior probability can be written as:

$$\log P(\boldsymbol{\theta}, \boldsymbol{\Lambda} \mid D) = \text{Constant} - (\frac{n}{2} + \nu_1 + 1)\log \sigma^2 - \frac{\nu_2}{\sigma^2} - \frac{1}{2\sigma^2}\sum_{t=1}^{N}(y_t - f(\mathbf{x}_t))^2 - \frac{1}{2}\sum_{i=0}^{p} I_{\alpha i}(\log {\sigma_\alpha}^2 + \frac{{\alpha_i}^2}{{\sigma_\alpha}^2})$$
$$- \frac{1}{2}\sum_{j=1}^{M} I_{\beta j}\Phi(\sum_{i=0}^{p} I_{\gamma_{ji}})(\log {\sigma_\beta}^2 + \frac{{\beta_j}^2}{{\sigma_\beta}^2}) - \frac{1}{2}\sum_{j=1}^{M}\sum_{i=0}^{p} I_{\beta_j} I_{\gamma_{ji}}(\log {\sigma_\gamma}^2 + \frac{{\gamma_{ji}}^2}{{\sigma_\gamma}^2}) - \frac{m}{2}\log(2\pi) + m\log\lambda - \log(m!) \quad (11)$$

In order to effectively implement the BNNs, the data preparation and hyperparameter settings suggested by Liang [2005b] were adopted in this study. Firstly, the input and output data were normalized by $q_t^{'} = (q_t - \bar{q})/S_q$, where $\bar{q}$ and S_q denote the mean and standard deviation of the input and output data. This type of data processing tends to avoid that the neural networks are trained to accommodate different scales of the observed data. For the settings of the hyperparameters, moderate values were adopted to penalize a large weight variation and complex model structure: ${\sigma_\alpha}^2$, ${\sigma_\beta}^2$, and ${\sigma_\gamma}^2$ are set to 5, and a vague prior on σ^2 was chosen through setting $\nu_1 = \nu_2 = 0.05$.

2.1.4 Evolutionary Monte Carlo

Many MCMC methods have been shown to be effective for training BNNs, such as hybrid Monte Carlo [e.g., Neal, 1996], reversible jump MCMC [e.g., Green, 1995, Andrieu et al., 2001], sequential Monte Carlo [e.g., de Freitas *et al.*, 2001, Higdon *et al.*, 2002], and evolutionary Monte Carlo [e.g., Liang, 2005b]. Also several other MCMC based algorithms have been successfully applied to generating variables observing some complex distributions in water resources modeling. For example, the Shuffled Complex Evolution Metropolis algorithm (SCEM) [Vrugt *et al.*, 2003] and adaptive Metropolis samplers [Haario *et al.*, 2001; Kingston *et al*, 2005; Marshall *et al.*, 2004; Renard *et al.*, 2006] have been successfully applied in hydrologic modeling. All these MCMC

algorithms can serve as useful tools for training the BNNs. Although the comparison of efficiency and effectiveness of these MCMC methods is an interesting topic, it is beyond the scope of this paper. In this study, the Evolutionary Monte Carlo (EMC) algorithm was applied to train the BNNs. The EMC algorithm has been compared with several other famous MCMC methods, including the Gibbs [Chib, 1995], reversible jump MCMC [Green, 1995], and parallel tempering [Geyer, 1991], and was shown to be a promising MCMC method [Liang and Wong, 2001c].

The EMC is a population-based method, developed based on the combination of three popular algorithms: parallel tempering, reversible jump MCMC, and the genetic algorithm [Holland 1975; Goldberg 1989]. EMC combines the strength of genetic algorithm for parameter optimization and the capacity of MCMC for generating samples observing target distribution [Liang and Wong, 2001]. EMC generates new samples using the basic mutation, crossover operators in genetic algorithm. The acceptance of new samples is guided by the Metropolis-Hastings rule [Metropolis *et al.*, 1953; Hastings, 1970]. In addition, EMC allows position exchanges between the candidates within the population. In the following sections, the basic EMC algorithm is introduced briefly. It is assumed that the researchers are interested in sampling from the distribution $f(\xi) \propto \exp(-H(\xi))$, where $H(\cdot)$ is called the energy function of ξ, which corresponds to the negative log-posterior of a posterior distribution. Here ξ is referred to as an individual in the population, and represents the joint of one model structure and the corresponding set of parameters. The EMC is running multiple chains of different density distributions conditioned on the temperatures. A population of distributions $f_1(\xi^1),\ldots,$ $f_N(\xi^N)$ are constructed as: $f_i(\xi^i) \propto \exp(-H(\xi^i)/t_i)$, $i=1,\ldots,N$, where t_i is called the

temperature of $f_i(\cdot)$, N is called the population size. $\mathbf{t}=(t_1,\ldots t_N)$ is a series of temperatures defined by the users with $(t_1>\cdots>t_N)$, and ξ^i denotes a sample from $f_i(\cdot)$. After randomly initializing a population of samples, the specific mutation, crossover, and exchange operators are implemented to update the position of samples. Although the mutation and crossover operators are similar to those applied in genetic algorithm, they have been modified such that they are reversible and usable as proposal functions for the Metropolis-Hastings algorithm [Liang and Wong, 2001a, 2001b, 2001c; Liang, 2005b]. A simple introduction of the three operators is referred in Appendix A. Figure 2 shows the schematic diagram of one iteration of the EMC algorithm. In one iteration of EMC, the individuals in the population are first updated using the mutation operator (with probability of η) and the crossover operator (with probability of 1-η). Then, the exchange operator is implemented to exchange the positions of $N-1$ pairs of randomly sampled individuals (ξ^i, ξ^j). In the implementation of mutation and crossover operators, new samples are generated and the model needs to be evaluated, while no new samples are yielded and model evaluation is not needed during the exchange operation. The EMC algorithm has several attractive properties for effectively and efficiently generating samples from the model space [Liang and Wong, 2001a, 2001b, 2001c; Liang, 2005b]: 1) adopting a sequence of distributions along a temperature series can help the sampler overcome barriers of the energy function landscape; 2) the crossover operator enables EMC to have the learning ability of the genetic algorithm; 3) the exchange operator accelerates the mix of individuals in different sequences without additional evaluation of the energy function.

In order to implement the EMC algorithm, the user needs to specify several parameters that control its effectiveness and efficiency. These parameters include: the population size N, mutation rate η, Metropolis step size κ, and temperature series $\boldsymbol{t}$. Based on the recommendation in Liang and Wong [2001c] and Liang [2005b], we set $N = 20$, $\eta = 0.6$, $\kappa = 0.25$, the highest temperature $t_1 = 20$, the lowest temperature $t_N = 1$, and the intermediate temperatures equally spaced in between t_1 and t_N. At the high temperature, the transition of the system is relatively easier. This helps explore the sampling space. At the low temperature, the movement of sampler is relatively slow, which helps exploit the sampling space**.** The step size was calibrated such that the resulting acceptance rate ranged from 0.2 to 0.4 as suggested by Gelman *et al.,* [1996]. See Liang and Wong [2001a, 2001b, 2001c] for more discussion on the settings of the parameters for EMC.

In implementing a MCMC method, it is important to check whether the sampler has converged to the target distribution or not. As the EMC is running with multiple chains of different distributions conditioned on a series of temperatures, it is difficult to apply the scale-reduction score to diagnose whether the MCMC sampler converges or not. In this study, one commonly used diagnostic method through trace plots of MCMC samples versus iteration was applied to detect the convergence of the EMC sampler. It is assumed that the convergence has been reached when the trace plot flattens out [Kass *et al.,* 1998]. With the multimodal nature of neural networks' weights and structures, the convergence to the posterior weights distribution is usually very slow.

2.2 Study area description

2.2.1 Reynolds Creek Watershed

The 239 km^2 Reynolds Creek Experimental Watershed (Figure 3) is located about 80 km southwest of Boise, ID and exhibits a considerable degree of spatial heterogeneity. The topography of the watershed ranges from a broad, flat alluvial valley to steep, rugged mountain slopes, with a range in elevation from 1101 to 2241 m [Seyfried *et al.*, 2000]. Because of orographic effects, the average annual precipitation ranges from about 250 mm near the outlet to more than 1100 mm at the upper end of the watershed. Perennial streamflow is generated at the highest elevations in the southern part of Reynolds Creek where deep, late-lying snowpacks are the source for most water [Seyfried *et al*, 2000]. Although much of the watershed has steep, shallow, rocky soils, there are areas of deep, loamy soils that are rock-free. Land cover on Reynolds Creek consists of rangeland and forest communities of sagebrush, greasewood, aspen, and conifers (94%), and irrigated cropland (6%).

2.2.2 Little River Watershed

The 334 km^2 Little River Experimental Watershed (Figure 3) has been the subject of long-term hydrologic and water quality research by USDA-ARS and cooperators [Sheridan, 1997]. The watershed is located in the Tifton Upland physiographic region, characterized by intensive agriculture in relatively small fields in upland areas and riparian forests along stream channels. The region has low topographic relief and is characterized by broad, flat alluvial floodplains, river terraces, and gently sloping uplands [Sheridan, 1997]. Climate in this region is characterized as humid subtropical with an

average annual precipitation of about 1167 mm based on data collected by USDA ARS from 1971 to 2000. Soils on the watershed are predominantly sands and sandy loams with high infiltration rates. Since surface soils are underlain by shallow, relatively impermeable subsurface horizons, deep seepage and recharge to regional ground water systems are impeded (Sheridan, 1997). Land use types include forest (65%), cropland (30%), rangeland and pasture (2%), wetland (2%), and miscellaneous (1%).

2.3 Evaluation of the performance of BNNs

In hydrologic modeling, different types of uncertainty limits can be recognized [e.g. Beven, 2006; Liu and Gupta, 2007]. In this study, we are concerned with the modeling uncertainty and predictive uncertainty [Liu and Gupta, 2007]. The modeling uncertainty limits, obtained through training BNNs to match observed streamflow data, were expected to include a specified proportion of the training data set. The predictive uncertainty limits, obtained through applying the trained models to another independent data set, were expected to contain a specified proportion of future observations. Ideally, the uncertainty interval should be consistent with observations and be as small as possible [Vrugt *et al.*, 2007]. Two coefficients were used to compare the uncertainty intervals estimated by different BNNs: 1) the percentage of coverage (POC) of observations in the uncertainty interval, and 2) the average width (AW) of the uncertainty interval. The POC coefficient is firstly evaluated. The uncertainty interval with a POC coefficient close to the expected proportion is preferred. If the POC of two uncertainty intervals are very close to each other, then the uncertainty interval with narrower AW value is considered better.

In this investigation, we are interested in the modeling and predictive uncertainty limits of streamflow simulation using BNNs with different treatment of uncertainties associated with parameters and structures of neural networks. Four types of BNNs were applied in this study. The first type of BNNs, referred to as BNN-a, is with a fixed model structure and non-informative prior knowledge of parameters. The second one is BNN-b, which uses variable model structure and non-informative prior knowledge of parameters and model structures. The form of the posterior distribution used by BNN-a and BNN-b is defined by equation (8). The other two BNNs are referred to as BNN-c and BNN-d, respectively. BNN-c is with fixed model structure and informative prior knowledge of parameters. BNN-d uses variable model structure and informative prior knowledge of parameters and model structures. The form of the posterior distribution used by BNN-c is a simplification of equation (9) through setting all the indicator functions equal to 1. The form of the posterior distribution used by BNN-d is defined by equation (11). Several comparison scenarios were designed to show the effect of taking variable model structure and informative prior knowledge into account on the uncertainty limits estimation using BNNs: 1) comparing the uncertainty limits obtained by BNN-a and BNN-b can provide insight into the effect of considering model structure uncertainty under the non-informative prior knowledge condition; 2) comparing BNN-c and BNN-d can show the effect of considering variable model structure under the informative prior knowledge condition; 3) comparing BNN-a and BNN-c can reveal the effect of considering prior knowledge under the fixed model structure condition; 4) comparing BNN-b and BNN-d can show the effect of considering prior knowledge under the variable model structure condition. These comparisons are expected to provide some insight into response of

uncertainty limits due to different considerations of uncertainties related to parameters and model structure.

For reference purposes, the generalization ability of the BNNs was compared with the deterministic ANNs. Two types of ANNs described in section 2.1.1 were applied in this study. The first one is based on equation (1) and the second one is based on equation (2). The first type of ANNs, referred to as ANN-1, is with fixed model structure, and the second type of ANNs, referred to as ANN-2, is with variable structure. The generalization ability of BNNs and traditional ANNs was compared using two coefficients, including R^2 (coefficient of determination) and MSE (mean square error) with unit of square cms (cubic meter per second).

3 Results and Discussions

3.1 Evaluation of EMC for two illustrative examples

Before running the EMC algorithm for training the BNNs, two illustrative test examples were used to show the effectiveness of EMC for generating samples from complex distributions.

3.1.1 Evaluation of EMC for a bimodal distribution

The first illustrative example is a mix of two multivariate normal distributions with mean vectors of $\mathbf{0} = (0,0,0,0,0)$ and $\mathbf{8} = (8,8,8,8,8)$ respectively,

$$\pi(\mathbf{x}) = 0.5 \cdot N(\mathbf{0}, \mathbf{I}_5) + 0.5 \cdot N(\mathbf{8}, \mathbf{I}_5) \qquad (12)$$

where $\mathbf{x} = (x_1, x_2, x_3, x_4, x_5)$ is a five dimensional vector and $N(\mathbf{u}, \mathbf{\Sigma})$ denotes the normal distribution with mean of $\mathbf{u}$ and covariance matrix $\mathbf{\Sigma}$. This example was used by

Renard *et al.* [2006] to test the effectiveness of three MCMC samplers. The initial population was generated within [0,1] for each dimension. 30,000 iterations of EMC were implemented to estimate the statistical properties of $\mathbf{x}$. The EMC was implemented 50 times. In order to save space, only the statistics of the first dimension and fifth dimension of $\mathbf{x}$ were listed in Table 1 and Figure 4. The results show that the EMC algorithm can accurately generate samples that represent this bimodal distribution.

3.1.2 Evaluation of EMC for a multimodal distribution

A two dimensional multimodal mixture normal distribution was used to test the EMC algorithm to show its effectiveness for sampling candidates from a distribution with a complex landscape. The multimodal distribution applied here is

$$f(\mathbf{x}) = \frac{1}{\sqrt{2\pi}\sigma}\sum_{i=1}^{20}\omega_i \exp\left\{-\frac{1}{2\sigma^2}(\mathbf{x}-\mathbf{u}_i)'(\mathbf{x}-\mathbf{u}_i)\right\} \qquad (13)$$

where σ=0.1, $\omega_1 = \cdots = \omega_{20}$=0.05. The mean vectors $u_1, u_2, \ldots u_{20}$ are randomly drawn from within [0,1] for each dimension. This multimodal distribution is similar to that used in Liang and Wong [2001c]. The initial population was generated within [0,1]. 50,000 iterations of EMC were implemented to estimate the statistical properties of $\mathbf{x}$. The EMC was implemented 50 times. Figure 5 shows the scatter plot of 10,000 samples, which reveals that the EMC can effectively sample all the 20 local modes. The estimate of means, variances of the two components of $\mathbf{x}$, and the standard deviation of the estimated values are shown in Table 2, which shows that the EMC algorithm can consistently obtain accurate estimates of the statistical properties of $\mathbf{x}$.

3.2 Application of BNNs for Streamflow Simulation in two experimental watersheds

3.2.1 Application of BNNs in the Reynolds Creek Experimental Watershed

Streamflow during low temperature periods (late fall, winter, and early spring) in the Reynolds Creek Experimental Watershed (RCEW) is mainly driven by snowmelt. Because simulation of the snow-driven flow during these low temperature periods requires long term climate inputs, stremflow simulation during these periods was not included in the data sets for this study. Streamflow data from day 148 to 274 for water years (WY) 1968-1975 (a total of 1016 data values) were used in this study. These 1016 data values were further subdivided into two groups. The first group included the streamflow data in WY 1973-1975, which were used for neural networks training. The second group included the streamflow data in WY 1968-1972, which were used to test the generalizing ability of trained networks. Model setup of the neural networks involves the selection of input variables and hidden units. The input variables for the first layer of a three-layer perceptron network were selected based on the knowledge of the hydrologic characteristics of the study area and the correlation between the input variables and streamflow data. A total of 13 input variables were selected: 1) total daily precipitation of the last four days starting from $t-1$ to $t-4$ was taken into account as four separate inputs; 2) moving average of the last 30 days' precipitation as a single separate input; 3) mean daily temperature of the last four days starting from $t-1$ to $t-4$ were taken as four separate inputs; 4) moving average of last 30 days' temperature as one input; 5)

daily streamflow values of the last three days from $t-1$ to $t-3$ were included according to the partial auto-correlation function (PACF). The selection of the number of hidden units was based on a trial and error procedure, and twenty hidden units were chosen for this case study. In the selection of hidden units, the traditional neutral networks with fixed model structure were applied. This number of hidden units was considered as adequate for the neural networks with variable structure. Further tests using larger number of hidden units did not show pronounced improvement for the network with variable structure. The EMC algorithm was implemented to train the ANNs and BNNs. For training the Bayesian neural networks, the EMC was run for 200,000 iterations each time. The trace of the mean log posterior density was inspected, and it was found that convergence was reached after about 100,000 iterations. For each run of EMC-based BNNs, the first 100,000 iterations were taken as a burn-in stage and were discarded, and 10,000 sets of $(\boldsymbol{\theta}, \boldsymbol{\Lambda})$ separated with equal interval were sampled from the remaining 100,000 iterations. A total of 50,000 samples was collected to derive the posterior distribution of $(\boldsymbol{\theta}, \boldsymbol{\Lambda})$, which were further used to run the neural network and calculate the network output $f(\mathbf{x}, \boldsymbol{\theta}, \boldsymbol{\Lambda})_k$ ($k = 1,2,\ldots,50000$). The mean of the simulations was used as the estimate of the streamflow value. All the 50,000 $f(\mathbf{x}, \boldsymbol{\theta}, \boldsymbol{\Lambda})$ predictions were ranked in ascending order to determine the 95% prediction interval. The two types of ANNs were also calibrated using the EMC algorithms with long iterations (1,000,000 iterations) to minimize the objective function (MSE).

The calibration and validation results of the ANNs and BNNs are listed in Table 3. For both calibration and validation periods, the ANN-2 performed better than ANN-1, which shows the advantage of variable structure over fixed structure neural networks.

Although the performance of deterministic ANNs is better than the four BNNs in the model training period, all four of the BNNs perform better than the ANNs during the validation period. The validation results provide valuable information regarding the model's performance [Khan and Coulibaly, 2006]. In the application of a model for streamflow prediction, we are more interested in the generalization ability of the model for data sets independent from calibration. For the validation period, the MSEs of ANN-1 and ANN-2 are 0.062 and 0.058 respectively, while the MSEs of BNN-a, BNN-b, BNN-c and BNN-d are 0.052, 0.05, 0.055 and 0.056, respectively. In terms of the MSE and R^2 coefficients for the validation period, the four BNNs show similar generalization performance with slight difference.

For the four types of BNNs, we are particularly interested in the modeling and predictive uncertainty limits. For illustrative purposes, the modeling uncertainty intervals estimated by different BNNs for days from May 28, 1975 to July 12, 1975 during calibration period are shown in Figure 6. Similarly the predictive uncertainty intervals for days from May 28, 1972 to June 28, 1972 during validation period are shown in Figure 7. From Figures 6 and 7, it is evident that the different BNNs yield various 95% uncertainty intervals. For example, the 95% uncertainty interval estimated by BNN-a is very narrow compared with other BNNs for both calibration and validation periods. The two coefficients (POC and AW) obtained by the four BNNs are shown in Table 3. For the calibration period, BNN-a, BNN-b, BNN-c, and BNN-d produce POC coefficients of 65.8%, 80%, 93.2%, and 93.7%, respectively. And for validation period, BNN-a, BNN-b, BNN-c, and BNN-d produce POC coefficients of 73.8%, 83.5%, 93.7%, and 94%, respectively. Using the comparison scenarios described in section 2.2, we evaluated the

95% uncertainty intervals estimated by the four BNNs to show the effect of different considerations of uncertainty sources: 1) BNN-b includes about 10% more observations than BNN-a for both modeling and predictive uncertainty limits, which shows that considering variable model structures can improve the estimation of uncertainty limits under the non-informative prior knowledge condition; 2) under the informative prior knowledge condition, BNN-d yields 95% modeling and predictive uncertainty intervals with slightly better POC coefficients than BNN-c, while the AWs of BNN-c are slightly smaller than BNN-d. In this case, variable model structures don not necessarily mean better uncertainty estimation of streamflow simulation 3) Under both fixed and variable model structure conditions, incorporating informative prior knowledge of uncertainty sources can improve the uncertainty interval estimation to include more observations. The BNN-c includes about 28% and 20% more observations than BNN-a for modeling and predictive uncertainty intervals, respectively. BNN-d includes 10% more observations than BNN-b for both calibration and validation periods. Through comparing the uncertainty intervals of the four BNNs, it is evident that the choice of posterior model probability with different considerations of the uncertainties associated with parameters and structures can exert appreciable effects on the estimated uncertainty interval.

3.2.2 Application of BNNs in the Little River Experimental Watershed

Streamflow data of WY 1997-2002 in the Little River Experimental Watershed (LREW) were used to develop and validate the BNNs and ANNs. These five years of daily data values were subdivided into two groups. The first group includes the streamflow values in WY 1997-2000, which were used for neural networks training. The second group includes the streamflow values in WY 2001-2002, which were used to test

the generalizing ability of trained networks. The procedures for model setup of the neural network for the LREW are similar to those used in the RCEW. A total of 15 input variables were identified: 1) daily precipitation of the last five days starting from $t-1$ to $t-5$ was taken into account as five separate inputs; 2) moving average of the last 30 days' precipitation as a single separate input; 3) mean daily temperature of the last four days starting from $t-1$ to $t-5$ were taken as five separate inputs; 4) moving average of the last 30 days' temperature as one input; 5) daily streamflow values of the last three days from $t-1$ to $t-3$ were included as three separate according to the partial auto-correlation function (PACF). Thirty hidden units were used to determine the neural network structure of the ANNs and BNNs. The EMC algorithms were used to generate 50,000 sets of $(\boldsymbol{\theta}, \boldsymbol{\Lambda})$, and estimate the 95% uncertainty intervals of the four types of BNNs.

The calibration and validation results of the ANNs and BNNs in the LREW are listed in Table 4. In terms of generalization capacity, the results obtained for the LREW are similar to those obtained for the RCEW. ANN-2 with variable structure performed better than ANN-1 with fixed structure. ANN-1 and ANN-2 performed better than the BNNs in the calibration period, while all four BNNs performed better than ANNs in the validation period. In addition, the generalization performance of the four BNNs is similar to each other.

Further analysis of the modeling and predictive uncertainty intervals estimated by the four BNNs in the LREW was also conducted. For illustrative purposes, the 95% modeling uncertainty intervals for days from January 4, 1997 to March 31, 1997 are shown in Figure 8 (calibration period), and the 95% predictive uncertainty intervals for

days from January13, 2001 to April 24, 2001 are shown in Figure 9 (validation period). Visually, the difference between the uncertainty intervals estimated by the four BNNs in the LREW is not so appreciable compared with that obtained in the RCEW. The POCs and AWs of the 95% uncertainty intervals estimated by different BNNs are listed in Table 4. For the calibration period, the BNN-a, BNN-b, BNN-c and BNN-d include about 79.7%, 82.3%, 85.1%, and 86% of the observed data into their 95% modeling uncertainty intervals respectively. For the validation period, the 95% predictive uncertainty intervals of the four BNNs tend to expand and include more observed data. BNN-a, BNN-b, BNN-c, and BNN-d cover approximately 87.3%, 90.6 %, 90.9 %, and 92.1% of the observed streamflow data within their 95% predictive uncertainty intervals, respectively. The comparisons between the uncertainty intervals estimated by different BNNs in the LREW also show that incorporating variable model structures and informative prior knowledge can provide more reasonable estimation of the uncertainty of streamflow simulation: 1) under both variable and fixed model structure conditions, taking informative prior knowledge into account results in more robust modeling and predictive uncertainty intervals for BNN-d and BNN-c than BNN-b and BNN-a, respectively; 2) under both non-informative and informative prior knowledge conditions, BNN-b and BNN-d contain more observations in the modeling and predictive uncertainty limits than BNN-a and BNN-c respectively. Results obtained for the LREW are similar to those obtained for the RCEW, except that BNN-d produces not only larger POCs but also smaller AWs than BNN-c. This finding emphasizes the importance of considering multiple model structures.

3.2.3 Effect of prior settings and number of hidden units on the performance of BNNs

For the BNNs, the choice of posterior model probability with different considerations of uncertainties associated with model structures and parameters can exert substantial impacts on both modeling and predictive uncertainty limits estimated by the BNNs. Based on the test results in the LREW and RCEW, BNN-a, which only considers parameter uncertainty with non-informative prior knowledge, performs the least among all the four BNNs. On the other hand, BNN-d, which considers both parameter and model structure uncertainties with informative prior knowledge, produces equivalent or better estimation of the 95% modeling and predictive uncertainty intervals compared to the other BNNs. In general, incorporating variable model structure and informative prior knowledge produces more reasonable uncertainty interval estimation. It is important to recognize that model structures and prior knowledge of neural networks' parameters applied in this study is not selected arbitrarily, but based on expert knowledge and experimental testing [Wang, 1995, Müler and Insua, 1998, Liang, 2005b]. The effect of number of networks' hidden units and prior variances of weights on uncertainty estimation of BNNs was examined. Table 5 shows the performance of BNNs with different number of hidden units. The performance of BNNs is relatively stable regarding the change of number of hidden units. The BNNs with 20, 30 and 50 hidden units exhibited very similar performance, while the models with 5 and 10 hidden units are slightly inferior to the others (Table 5). The effect of prior settings on the performance of BNNs was also examined (Table 6). For medium prior variances of 5 and 10, the performance of BNNs is similar to each other. The performance of BNNs is deteriorated

when small and large prior variances (i.e. 1 and 100) of weights were used. With a prior variance of 100, the BNNs can only include less than 80% observations. These results are similar to those obtained by Liang (2005b). We also set the prior variance to IG (0.05, 0.5) distribution, which is similar to that used by Neal (1996). This hierarchical setting allows the variance to be determined from the data. The performance of hierarchical hyperprior is very similar to that of BNNs with fixed prior variances of 5 and 10 (Table 6). The above results indicate that inappropriate setting of model structure and prior knowledge may lead to worse estimation results.

3.2.4 Discussion

As shown in section 3.2.1 and 3.2.2, the Bayesian-based neural networks can produce more accurate prediction of daily streamflow than the deterministic ANNs in the validation period. Taking the uncertainty of model structures or parameters into account and applying the Bayesian model averaging scheme to estimate the network output can provide more reliable prediction than using a single model that is the best fit to the training data. This result is consistent with those obtained by Kingston *et al.* [2005] and Khan and Coulibaly [2006]. In addition to generalization ability, another advantage of the BNNs over the deterministic ANNs is that they can produce the uncertainty intervals that indicate the level of uncertainty in the forecasts.

Through incorporating variable model structure and prior knowledge into account, BNN-d obtained better estimation of 95% uncertainty interval, but it is worth noting that the probabilistic predictions at other coverage levels are also of interest (Krzysztofowicz, 2001; Montanari, 2005; Laio et al., 2007b). For example, Laio and Tamea (2007a) illustrated application of probabilistic prediction and cost-loss analysis in hydrologic

prediciton. In this study, the POC values obtained by BNN-d at different uncertainty intervals were calculated (Table 7). In both LREW and RCEW, the estimated uncertainty intervals do not include the corresponding expected percentage of observations. In the LREW, for uncertainty intervals with coverage level larger than 50%, the difference between estimated POC and expected coverage percentage is less than or close to 10%, while this difference is usually larger than 20% for uncertainty intervals with expected coverage level less than 50%. The maximum difference reached 31% for the 20% uncertainty interval in the validation period. In the RCEW, for uncertainty intervals with coverage level larger than 70%, the difference between POC and expected coverage percentage is less than 10%, while this difference is larger than 20% for uncertainty intervals with coverage level less than 70%. The maximum difference reached 38% for the 10% uncertainty interval in the validation period. Overall, the BNNs produced better approximation of uncertainty intervals with high coverage level than those with low coverage level. Further analysis of the POCs of rising and peak components of hydrograph was conducted, since these two parts are often of major interest to water management managers. From Figures 6-9, it seems that the 95% uncertainty intervals did not perform well for the rising and peak components of the hydrograph. In the LREW, the POC values of the rising and peak components of the hydrograph are 66% and 69% for calibration and validation periods, respective. In the RCEW, the POC values are 90% and 91% for calibration and validation periods, respectively. This indicates that BNNs exhibited various performances for different components of the hydrograph. There are several potential reasons for the inadequate performances of BNNs to capture the uncertainty intervals at different coverage levels and for different flow components. The

major reason is because our understanding of hydrologic uncertainty is still far from complete. The inappropriate convergence to the true posterior [Kingston *et al.*, 2005], the inadequate definition of the prior distribution of parameters and model structures, and the omission of uncertainties related to the observed input data and other forcing data can lead to inappropriate estimation of the uncertainties. Moreover, the complex and true joint distributions of the uncertainty sources (which result from high non-linearity of the hydrologic system and the complex interactions between different components of the system) make it very difficult to accurately represent the uncertainty of streamflow simulation [Liu and Gupta, 2007].

For water resources investigations essential for relevant decision making processes, the predictive uncertainty estimation of hydrologic prediction is valuable. The predictive uncertainty limits are dependent on and different from modeling uncertainty. This is because when the trained BNNs are applied to another set of data independent of the training data, the hydrologic conditions may change and therefore impact the predictive interval estimation. From Figures 6-9 and Tables 3-4, it can be seen that the 95% modeling uncertainty limits are always narrower than the corresponding predictive uncertainty limits estimated by the same BNNs. The difference between modeling and predictive uncertainty limits can be impacted by the type of BNN and the characteristics of the hydrologic conditions. For example, in the RCEW, the BNN-a's POC of predictive uncertainty interval (73.8%) is about 8% higher than its POC of modeling uncertainty interval (65.8%), while the BNN-d's POC of predictive uncertainty interval (94%) is about the same its POC of modeling uncertainty interval (93.7%). Applying the BNN-d to the LREW, it is apparent that the BNN-d's POC of predictive uncertainty interval

(92.1%) is 6% higher than its POC of modeling uncertainty interval (86%). Because of the future uncertainties due to natural and anthropogenic factors, the predictive uncertainty limits are also uncertain, which means that we are unable to estimate predictive uncertainty limits even if our estimation of modeling uncertainty limits are accurate. Hence in application of uncertainty analysis for hydrologic prediction, how to extend modeling uncertainty limits to predictive uncertainty limits remains a huge challenge for applying BNNs to water resources-related management and design problems.

Although uncertainty estimation of hydrologic prediction faces many challenges, it is still broadly recognized that proper consideration of uncertainty in hydrologic predictions is essential for purposes of both research and operational modeling [Wagener and Gupta, 2005; Pappenberger and Beven, 2006; Liu and Gupta, 2007]. To improve the estimation of modeling uncertainty of hydrologic modeling, effective methods for considering the uncertainties associated with input hydrometeorolgic data [e.g. Kavetshi *et al*, 2006, Ajami *et al.*, 2007, Srivastav *et al.*, 2007] and observed outputs [e.g. Kuczera, 1983; Bates and Campbell, 2001; Yang *et al.*, 2007] must also be considered in the definition of posterior model probability

4. Conclusions

Application of Bayesian neural networks is relatively new in hydrologic modeling. One preferred advantage of BNNs over the traditional deterministic ANNs is that BNNs can estimate the uncertainty of hydrologic prediction. Reasonable estimates of the predictive uncertainty of hydrologic simulation are critical for decision making in problems related to resources management. In this study, the BNNs were applied to the

Reynolds Creek and Little River Experimental Watersheds for daily streamflow simulation to examine the effect of different considerations of variable model structures and prior knowledge on the uncertainty limits estimation of BNNs. A major challenge facing the application of the BNNs is the effectiveness of the MCMC sampling algorithm for generating models (defined by the neural network's parameters and structures) observing the complex posterior distribution. An advanced MCMC sampler – EMC was tested for estimation of the statistical characteristics of variables observing complex multimodal distributions and then applied to train the BNNs.

Four types of BNNs with different treatments of variable structures and prior knowledge have been applied in this study. All four BNNs exhibited superior generalization capacity to the deterministic ANNs, which emphasize the prospect of BNNs in future hydrologic modeling. Findings from this study show that the 95% uncertainty limits of neural network outputs estimated by different BNNs were evidently different from each other. In general, BNNs incorporating multiple model structures can provide equal or better estimation of the uncertainty limits than those with fixed network structures. Findings also show that taking informative prior knowledge of network parameters and structures can lead to more robust estimation of the uncertainty limits. For all the test cases, the 95% uncertainty intervals (including modeling and predictive uncertainty intervals) estimated by all four BNNs failed to include 95% or more of observed streamflow data. Examination of the uncertainty intervals at different coverage levels and for different flow components also shows the inadequate performance of the BNNs. This, to some extent, indicates the incomplete consideration of all uncertainty sources and inappropriate definition of error characteristics associated with different

uncertainty sources. In the future, improving understanding and quantifying methods of different uncertainty sources need to be exploited for effective estimation of the uncertainty of hydrologic prediction using BNNs. It should also be noted that the difference between predictive uncertainty and modeling uncertainty, which is raised by unknown future conditions, complicates the process to develop practical guides on how to extend modeling uncertainty estimation to reliable predictive uncertainty estimation.

Acknowledgements

The authors would like to thank Dr. Amilcare Porporato and two anonymous reviewers for the excellent comments and suggestions, which greatly enhanced the quality of the manuscript.

Appendix A

A.1. *Mutation*

The mutation operator is used to generate the variant of a chromosome (denoted as ξ^i, where the superscript i is the position of a chromosome in the current population). ξ^i is selected at random from the current population $\mathbf{z} = \{\xi^1, \ldots, \xi^{i-1}, \xi^i, \xi^{i+1}, \ldots, \xi^N\}$. ξ^i is modified to form a new chromosome $\xi^{i'}$ by one of the three types of operations: "birth," "death," and "Metropolis." "Birth" is used to add effective connections to the neural network, while "death" is used to remove effective connections. "Metropolis" is the same operation defined by Metropolis et al. [1953] and Hasting [1970]. The newly generated population $\mathbf{z}' = \{\xi^1, \ldots, \xi^{i-1}, \xi^{i'}, \xi^{i+1}, \ldots, \xi^N\}$ is accepted with probability,

$$\min\left\{1, \exp\left\{-(H(\xi^{i'}) - H(\xi^i)/t_i)\right\}\frac{T(\mathbf{z}\,|\,\mathbf{z}')}{T(\mathbf{z}'|\,\mathbf{z})}\right\}$$

where $T(\mathbf{z}\,|\,\mathbf{z}')/T(\mathbf{z}'|\,\mathbf{z})$ is the ratio of the transition probabilities. For detailed information on the "birth" and "death" operations and calculation of the three types of transition probabilities, please refer to Liang and Wong [2001a, 2001b, 2001c] and Liang [2005b].

A.2. *Crossover*

The crossover operator is similar to that used in the popular Genetic Algorithm. Through recombination of two chromosomes, which are randomly selected from the current population, offspring are produced. First of all, two chromosomes ξ^i and ξ^j (j is the position of a chromosome in the current population with a different value from i) are

selected as parental chromosomes. Next, an integer c is drawn randomly among $\{1,2,\ldots,M\}$, where M is the number of hidden units. The hidden unit c is called the crossover unit, and only one unit crossover operator is applied in this study. Finally, two new offspring $\xi^{i'}$ and $\xi^{j'}$ are constructed by swapping the weights connected with hidden unit c between ξ^i and ξ^j. A new population is constructed by replacing the parental chromosomes with the new offspring, and it is accepted with probability

$$\min\left\{1, \exp\left\{-(H(\xi^{i'})-H(\xi^i))/t_i-(H(\xi^{j'})-H(\xi^j))/t_j\right\}\frac{T(\mathbf{z}\,|\,\mathbf{z}')}{T(\mathbf{z}'|\,\mathbf{z})}\right\}$$

where $T(\mathbf{z}'|\,\mathbf{z})=P(\xi^i,\xi^j\,|\,\mathbf{z})P(\xi^{i'},\xi^{j'}\,|\,\xi^i,\xi^j)$, $P(\xi^i,\xi^j\,|\,\mathbf{z})=2/(N(N-1))$ is the selection probability of (ξ^i,ξ^j) from population $\mathbf{z}$, and $P(\xi^{i'},\xi^{j'}\,|\,\xi^i,\xi^j)$ denotes the generating probability of $(\xi^{i'},\xi^{j'})$ from the parental chromosomes (ξ^i,ξ^j). The crossover operator is symmetric, which means that $P(\xi^{i'},\xi^{j'}\,|\,\xi^i,\xi^j) = P(\xi^i,\xi^j\,|\,\xi^{i'},\xi^{j'})$. $T(\mathbf{z}\,|\,\mathbf{z}')/T(\mathbf{z}'|\,\mathbf{z})=1$ for the crossover operator.

A.3. ***Exchange***

The exchange operator is useful for exchangimg information obtained by different series of chromosomes within the population. Given the current population $\mathbf{z}$ and the attached temperature ladder $\mathbf{t}$, an exchange is made between ξ^i and ξ^j without changing the temperature t associated with the specific position within the population. The initial population and temperature ladder $(\mathbf{z},\mathbf{t})=(\xi^1,t_1,\ldots,\xi^i,t_i,\ldots,\xi^j,t_j,\ldots,\xi^N,t_N)$ are proposed to be changed to $(\mathbf{z}',\mathbf{t})=(\xi^{1'},t_1,\ldots,\xi^{i'},t_i,\ldots,\xi^{j'},t_j,\ldots,\xi^{N'},t_N)$. In this paper, the exchange is only operated on two chromosomes neighboring each other (i.e., $|i-j|=1$). The new population is accepted with probability

$$\min\left\{1, \exp\left\{(H(\xi^i) - H(\xi^j))\left(\frac{1}{t_i} - \frac{1}{t_j}\right)\right\}\frac{T(\mathbf{z}\,|\,\mathbf{z}')}{T(\mathbf{z}'|\,\mathbf{z})}\right\}$$

Where $T(\mathbf{z}'|\,\mathbf{z}) = p(\xi^i)w_{i,j} + p(\xi_j)w_{j,i}$, $P(\xi^i)$ is the probability that ξ^i is chosen to exchange with the other chromosome, $w_{i,j}$ denotes the probability that ξ^j is chosen to exchange with ξ^i, $w_{i,i+1} = w_{i,i-1} = 0.5$ for $1 < i < N$ for and $w_{1,2} = w_{N,N-1} = 1$. Thus $T(\mathbf{z}\,|\,\mathbf{z}')/T(\mathbf{z}'|\,\mathbf{z}) = 1$ for the exchange operator.

References

Ajami, N. K., Q. Duan, and S. Sorooshian (2007), An integrated hydrologic Bayesian multimodel combination framework: Confronting input, parameter, and model structural uncertainty in hydrologic prediction, *Water Resour. Res*., 43, W01403, doi:10.1029/2005WR004745.

Andrieu, C., N. D. Freitas, and A. Doucet (2001), Robust full Bayesian learning for radial basis networks, *Neural Comp.,* 13, 2359-2407.

ASCE Task Committee on Application of Artificial Neural Networks in Hydrology (2000a), Artificial neural networks in hydrology. I: Preliminary concepts, *J. Hydrol. Eng.*, 5(2), 115–123.

ASCE Task Committee on Application of Artificial Neural Networks in Hydrology (2000b), Artificial neural networks in hydrology. II: Hydrologic applications, *J. Hydrol. Eng.*, 5(2), 124–137.

Bates, B. C., and E. P. Campbell (2001), A Markov chain Monte Carlo scheme for parameter estimation and inference in conceptual rainfall-runoff modeling, *Water Resour. Res.*, *37*(4), 937–947.

Beven, K. J. (2006), A manifesto for the enquiringly thesis, *J. Hydrol.,* 320, 18-36.

Beven, K. J., and J. Freer (2001), Equifinality, data assimilation, and uncertainty estimation in mechanistic modeling of complex environmental systems, *J. Hydrol.*, 249, 11-29.

Beven, K., and A. Binley (1992), The future of distributed models: Model calibration and uncertainty prediction, *Hydrol. Process.,* 6, 279-298.

Bishop, C. M. (1995), *Neural Networks for Pattern Recognition*, Oxford Univ. Press: NY.

Chetan, M., and K. P. Sudheer (2006), A hybrid linear-neural model for river flow forecasting, *Water Resour. Res.*, *42*, W04402, doi:10.1029/2005WR004072.

Chib, S. (1995), Marginal likelihood from the Gibbs output, *J. Amer. Stat. Associa.*, 90, 1313-1321.

Coulibaly, P., B. Bobe´e, and F. Anctil (2001b), Improving extreme hydrologic events forecasting using a new criterion for artificial neural network selection, *Hydrol. Process.*, 15, 1533–1536.

Dawson, C. W., and R. L. Wilby (2001), Hydrological modeling using artificial neural networks, *Prog. Phys. Geogr.*, 25, 80-108.

de Freitas J. F. G., M. Niranjan, A. H. Gee, and A. Doucet (2000), Sequential Monte Carlo methods to train neural network models, 12, 955-993. doi:10.1029/2006WR005352.

Duan, Q., N. K. Ajami, X. Gao, and S. Sorooshian (2007), Multi-model ensemble hydrologic prediction using Bayesian model averaging, *Adv. Water Resour.*, 30(5), 1371-1386.

Gelman, A., R. O. Roberts, and W. R. Gilks (1996), Efficient Metropolis jumping rules, In Bayesian Statistics 5, Bernardo, J. M., J. O. Berger, A. P. Dawid, and A. F. M. Smith (Eds), Oxford Univ. Press: NY.

Georgakakos, K. P., D. J. Seo, H. Gupta, J. Schake, and M. B. Butts (2004), Characterizing streamflow simulation uncertainty through multimodel ensembles, *J. Hydrol.*, 298, 222-241.

Geyer, C. J. (1991), Markov chain Monte Carlo maximum likelihood, in *Computing Science and Statistics: Proceedings of the 23rd Symposium on the Interface*, ed. Keramigas E. M., Fairfax, VA: Interface Foundation, pp. 153–163.

Goldberg, D. (1989), *Genetic Algorithms in Search, Optimization and Machine Learning*, Addison-Wesley, Reading, MA.

Green, P. J. (1995), Reversible jump Markov chain Monte Carlo computation and Bayesian model determination, *Biometrika*, 82, 711–732.

Haario, H., E. Saksman, and J. Tamminen (2001). An adaptive metropolis algorithm, *Bernoulli*, *7*(2), 223–242.

Hastings, W. K. (1970), Monte-Carlo sampling methods using Markov Chains and their applications, *Biometrika*, 57, 97-109.

Higdon, D., H. Lee, and Z. Bi (2002), A Bayesian approach to characterizing uncertainty in inverse problems using coarse and fine-scale information, *IEEE Transactions on Signal Processing*, 50, 389-399.

Holland, J. (1975), *Adaptation in Natural and Artificial Systems*, University of Michigan Press: Ann Arbor, MI.

Jain, A., and S. Srinivasulu (2004), Development of effective and efficient rainfall-runoff models using integration of deterministic, real-coded genetic algorithms and artificial neural network techniques, *Water Resour. Res.*, *40*, W04302, doi:10.1029/2003WR002355.

Kavetski, D., G. Kuczera, and S. W. Franks (2006), Bayesian analysis of input uncertainty in hydrological modeling: 1. Theory, *Water Resour. Res.*, *42*, W03407, doi:10.1029/2005WR004368.

Keskin M. E., and ö. Terzi (2006), Artificial neural network models of daily pan evaporation, *J. Hydrol. Eng.*, 11(1): 65-70.

Khan, M. S., and P. Coulibaly (2006), Bayesian neural network for rainfall-runoff modeling, *Water Resour. Res*., 42, W07409, doi:10.1029/2005WR003971.

Kingston, G. B., M. F. Lambert, and H. R. Maier (2005), Bayesian training of artificial neural networks used for water resources modeling, *Water Resour. Res.*, *41*, W12409, doi:10.1029/2005WR004152.

Krzysztofowicz, R. (2001), The case for probabilistic forecasting in hydrology, *J. Hydrol.*, 249:2-9.

Kuczera, G. (1983), Improved parameter inference in catchment models: 1. Evaluating parameter uncertainty, *Water Resour. Res.*, 19(5), 1151-1162.

Laio F. and S. Tamea (2007a), Verification tools for probabilistic forecasts of continuous hydrological variables, *Hydrol. Earth Syst. Sci.*, 11: 1267-1277.

Laio, F. L. Ridolfi, and S. Tamea (2007b). Probabilistic prediction of real-world time series: a local regression approach, *Geophys. Res. Lett.*, 34, L03403,doi:10.1029/2006GL028776.

Lampinen, J., and A. Vehtari (2001), Bayesian approach for neural networks – reviews and case studies, *Neural Networks*, 14(3), 7-24.

Liang, F. (2003), An effective Bayesian neural network classifier with a comparison study to support vector machine, *Neural Comp.*, 15, 1959-1989.

Liang, F. (2005a), Evidence evaluation for Bayesian neural networks, *Neural Comp.*, 17, 1385-1410.

Liang, F. (2005b), Bayesian neural networks for non-linear time series forecasting, *Statis. Comp.*, 15, 13-29.

Liang, F. and Y. C. A. Kuk (2004), A finite population estimation study with Bayesian neural networks, *Survey Methodology*, 30, 219-234.

Liang, F., and W.H. Wong (2001a). Evolutionary Monte Carlo for protein folding simulations, *J. Chem Phys*., 115, 3374-3380.

Liang, F., and W.H. Wong (2001b), Evolutionary Monte Carlo sampling: applications to C_p model sampling and change-point problem, *Statistica Sinica*,10, 317-342.

Liang, F., and W.H. Wong (2001c), Real-parameter evolutionary sampling with applications in Bayesian Mixture Models, *J. Amer. Statist. Asso*c., 96, 653-666.

Liu, Y., and H. V. Gupta (2007), Uncertainty in hydrologic modeling: Toward an integrated data assimilation framework, *Water Resour. Res.*, 43, W07401, doi:10.1029/2006WR005756.

MacKay D. J. C. (1992), A practical Bayesian framework for backprop networks, *Neural Comp.*, 4, 448–472.

Maier, H. R., and G. C. Dandy (2000), Neural networks for the prediction and forecasting of water resources variables: A review of modeling issues and applications, *Environ. Model. Software*, 15, 101-123.

Marshall, L., D. Nott, and A. Sharma (2004), A comparative study of Markov chain Monte Carlo methods for conceptual rainfall-runoff modeling, *Water Resour. Res.*, 40, W02501, doi:10.1029/2003WR002378.

Metropolis, N., A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller (1953), Equation of state calculations by fast computing machines, *J. Chem. Phys.*, 21, 1087-1091.

Montanari, A. (2005), Large sample behaviors of the generalized likelihood uncertainty estimation (GLUE) in assessing the uncertainty of rainfall-runoff simulations, *Water Resour. Res.*, 41, W08406, doi:10.1029/2004WR003826, 2005.

Müller P., and D.R. Insua (1998), Issues in Bayesian analysis of neural network models. *Neural Comp.*, 10, 749–770.

Nayak, P. C., K. P. Sudheer, and S. K. Jain (2007), Rainfall-runoff modeling through hybrid intelligent system, *Water Resour. Res.*, *43*, W07415, doi:10.1029/2006WR004930.

Neal, R. M. (1996), *Bayesian Learning for Neural Networks*, New York: Springer-Verlag.

Pappenberger, F., and K. J. Beven (2006), Ignorance is bliss: Or seven reasons not to use uncertainty analysis, *Water Resour. Res.*, *42*, W05302, doi:10.1029/2005WR004820.

Parasuraman, K., A. Elshorbagy, and S. K. Carey (2006), Spiking modular neural networks: A neural network modeling approach for hydrological processes, *Water Resour. Res.*, *42*, W05412, doi:10.1029/2005WR004317.

Raghuwanshi, N. S., R. Singh, and L. S. Reddy (2006). Runoff and sediment yield modeling using artificial neural networks: Upper Siwane River, India, *J. of Hydrol. Eng.*, 11(1), 71-79.

Renard, B., V. Garreta, and M. Lang (2006), An application of Bayesian analysis and Markov chain Monte Carlo methods to the estimation of a regional trend in annual maxima, *Water Resour. Res.*, *42*, W12422, doi:10.1029/2005WR004591.

Sahoo, G. B., C. Ray, and E. H. De Carlo (2006). Use of neural network to predict flash flood and attendant water qualities of a mountainous stream on Oahu, Hawaii, *J. Hydrol.*, 327, 525-538.

Seidou, O., T. B. M. J. Ouarda, L. Bilodeau, M. Hessami, A. St-Hilaire, and P. Bruneau (2006), Modeling ice growth on Canadian lakes using artificial neural networks, *Water Resour. Res.*, *42*, W11407, doi:10.1029/2005WR004622.

Seyfried, M.S., R.C. Harris, D. Marks, and B. Jacob (2000), *A geographic database for watershed research, Reynolds Creek Experimental Watershed, Idaho, USA*, USDA ARS Technical Bulletin NWRC-2000-3.

Sheridan, J. M. (1997), Rainfall-streamflow relations for coastal plain watersheds, *Trans. ASAE*, 13(3), 333-344.

Srivastav, R. K., K. P. Sudheer, and I. Chaubey (2007), A simplified approach to quantifying predictive and parametric uncertainty in artificial neural network hydrologic models, *Water Resour. Res.*, *43*, W10407,

Vrugt, J. A., and B. A. Robinson (2007), Treatment of uncertainty using ensemble methods: Comparison of sequential data assimilation and Bayesian model averaging, *Water Resour. Res.*, 43, W01411, doi:10.1029/2005WR004838.

Vrugt, J. A., H. V. Gupta, W. Bouten, and S. Sorooshian (2003), A Shuffled Complex Evolution Metropolis algorithm for optimization and uncertainty assessment of hydrologic model parameters, *Water Resour. Res.*, *39*(8), 1201, doi:10.1029/2002WR001642.

Wagener, T., and H. V. Gupta (2005), Model identification for hydrological forecasting under uncertainty, *Stoch. Environ. Res. Risk. Assess.,* 19, doi:10.1007/s00477-005-0006-5.

Wang, Y. (1995). Unpredictability of standard back propagation neural networks, *Manage. Sci.,* 41, 555-559.

Yang, J., P. Reichert, and K. C. Abbaspour (2007), Bayesian uncertainty analysis in distributed hydrologic modeling: A case study in the Thur River basin (Switzerland), *Water Resour. Res.,* doi:10.1029/2006WR005497, in press.

List of Tables

Table 1. Parameters estimation of the five-dimensional bimodal distribution.

Table 2. Parameters estimation of the two-dimensional multimodal distribution.

Table 3. Evaluation of the performance of the ANNs and BNNs for streamflow simulation in the Reynolds Creek Watershed.

Table 4. Evaluation of the performance of BNNs and ANNs for streamflow simulation in the Little River watershed.

Table 5. Effect of number of hidden units on the performance of BNNs.

Table 6. Effect of prior settings on the performance of BNNs.

Table 7. Percentage of coverage values of uncertainty intervals at different coverage level.

Table 1. Parameter estimation of the five-dimensional bimodal distribution

Parameter	True value	Estimate	SD
u_1	4	4.02	0.034
u_5	4	4.014	0.035
Σ_{11}	17	16.989	0.033
Σ_{55}	17	16.99	0.035
Σ_{15}	16	15.993	0.028

NOTE: Here u_1 and u_5 are the first and fifth component of the mean of $\mathbf{x}$; Σ_{11}, Σ_{55} and Σ_{15} are the variances and covariance of the first and second component of $\mathbf{x}$; SD denotes the standard deviation of the corresponding estimate.

Table 2. Parameter estimation of the two-dimensional multimodal distribution

Parameter	True value	Estimate	SD
u_1	5.123	5.128	0.015
u_2	5.093	5.089	0.021
Σ_{11}	5.623	5.627	0.032
Σ_{22}	8.641	8.648	0.036
Σ_{12}	-1.579	-1.583	0.025

NOTE: Here u_1 and u_2 are the first and second component of the mean of $\mathbf{x}$; Σ_{11}, Σ_{22} and Σ_{12} are the variances and covariance of the first and second component of $\mathbf{x}$; SD denotes the standard deviation of the corresponding estimate.

Table 3. Evaluation of the performance of the ANNs and BNNs for streamflow simulation in the Reynolds Creek Experimental Watershed

Period/Model \ Evaluation Coefficients		MSE	R^2	Percentage of coverage	Average interval width
Calibration	BNN-a	0.0055	0.98	65.8%	0.05
	BNN-b	0.0046	0.99	80%	0.08
	BNN-c	0.0058	0.98	93.2%	0.16
	BNN-d	0.0061	0.98	93.7%	0.17
	ANN-1	0.0051	0.98	-	-
	ANN-2	0.0045	0.99	-	-
Validation	BNN-a	0.0052	0.98	73.8 %	0.06
	BNN-b	0.0050	0.98	83.5%	0.08
	BNN-c	0.0055	0.98	93.7%	0.14
	BNN-d	0.0056	0.98	94%	0.15
	ANN-1	0.0062	0.98	-	-
	ANN-2	0.0058	0.98	-	-

Table 4. Evaluation of the performance of BNNs and ANNs for streamflow simulation in the Little River Experimental Watershed

Period/Model \ Evaluation Coefficients		MSE	R^2	Percentage of coverage	Average interval width
Calibration	BNN-a	11.25	0.94	79.7%	5.7
	BNN-b	11.17	0.94	82.3%	5.99
	BNN-c	12.16	0.93	85.1%	6.88
	BNN-d	13.14	0.93	86%	6.62
	ANN-1	11.18	0.94	-	-
	ANN-2	9.96	0.95	-	-
Validation	BNN-a	6.16	0.89	87.4%	5.32
	BNN-b	5.78	0.90	90.6%	5.59
	BNN-c	5.92	0.89	90.9%	6.58
	BNN-d	5.96	0.89	92.1%	5.8
	ANN-1	7.71	0.87	-	-
	ANN-2	7.2	0.88	-	-

Table 5. Effect of number of hidden units on the performance of BNNs.

Number of hidden units	LREW				RCEW			
	Calibration		Validation		Calibration		Validation	
	MSE	POC	MSE	POC	MSE	POC	MSE	POC
5	14.1	81.7%	6.84	86.2%	0.0066	88.4%	0.0057	89.5%
10	13.43	83%	6.47	90.6%	0.0065	91.3%	0.0058	92.4%
20	13.2	85.6%	5.85	91.1%	0.0061	93.7%	0.0056	94%
30	13.14	86%	5.96	92.1%	0.0058	93.5%	0.0057	94.1%
50	12.94	85.6%	6.18	90.6%	0.0056	93.8%	0.0061	93.6%

Table 6. Effect of prior settings on the performance of BNNs.

Prior settings	LREW				RCEW			
	Calibration		Validation		Calibration		Validation	
	MSE	POC	MSE	POC	MSE	POC	MSE	POC
1	15.86	84.2%	6.59	87.1%	0.0064	91.6%	0.0059	90.3%
5	13.17	86%	6.12	91.7%	0.0061	93.7%	0.0056	94%
10	13.42	84%	6.04	90.5%	0.0063	92.8%	0.0062	94.5%
100	14.5	71.1%	6.46	76.1%	0.0068	76.5%	0.0062	75.9%
Hierarchical	13.52	85.5%	6.22	90.8%	0.0066	92.2%	0.0061	94.4%

Table 7. Percentage of coverage values of uncertainty intervals at different coverage level.

Uncertainty interval		10%	20%	30%	40%	50%	60%	70%	80%	90%	95%
LREW	calibration	32%	38%	53%	58%	59%	63%	74%	83%	84%	86%
	validation	37%	51%	57%	64%	68%	75%	76%	84%	89%	92%
RCEW	calibration	39%	45%	59%	66%	75%	83%	86%	88%	90%	94%
	validation	48%	53%	64%	71%	77%	84%	87%	89%	91%	94%

List of Figures

Figure 1. A fully connected one-hidden-layer feed-forward neural network with four input units, four hidden units, and one output unit.

Figure 2. Schematic illustration of one iteration of EMC.

Figure 3. The geographic locations of the Reynolds Creek watershed and Little River watershed.

Figure 4. Simulated values of the first and fifth component from the two-modal distribution. The solid line is the true value, and the grey area is the density estimated by EMC.

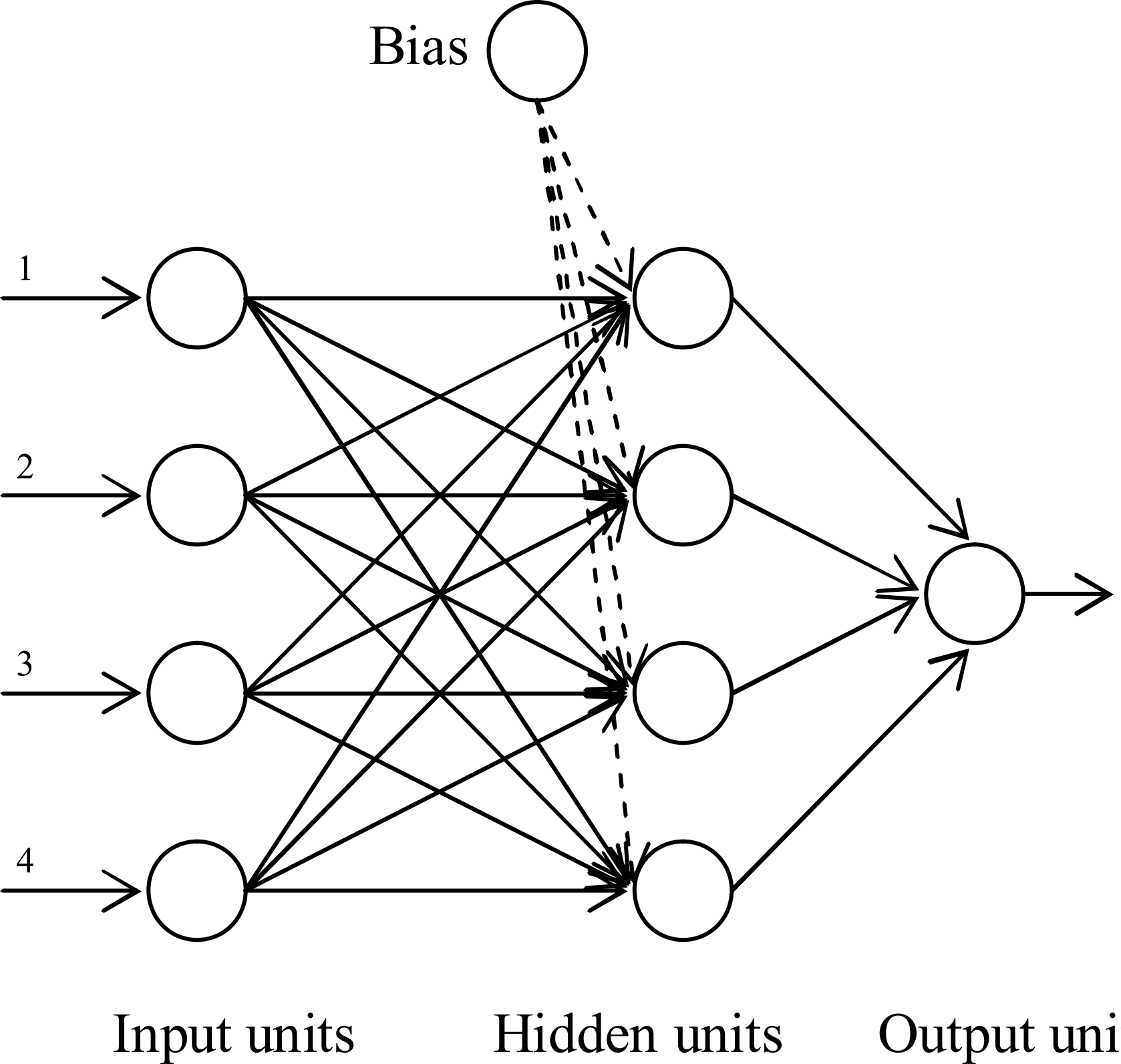
Figure 5. Scatter plot of the samples generated by EMC for the 20-modal distribution.

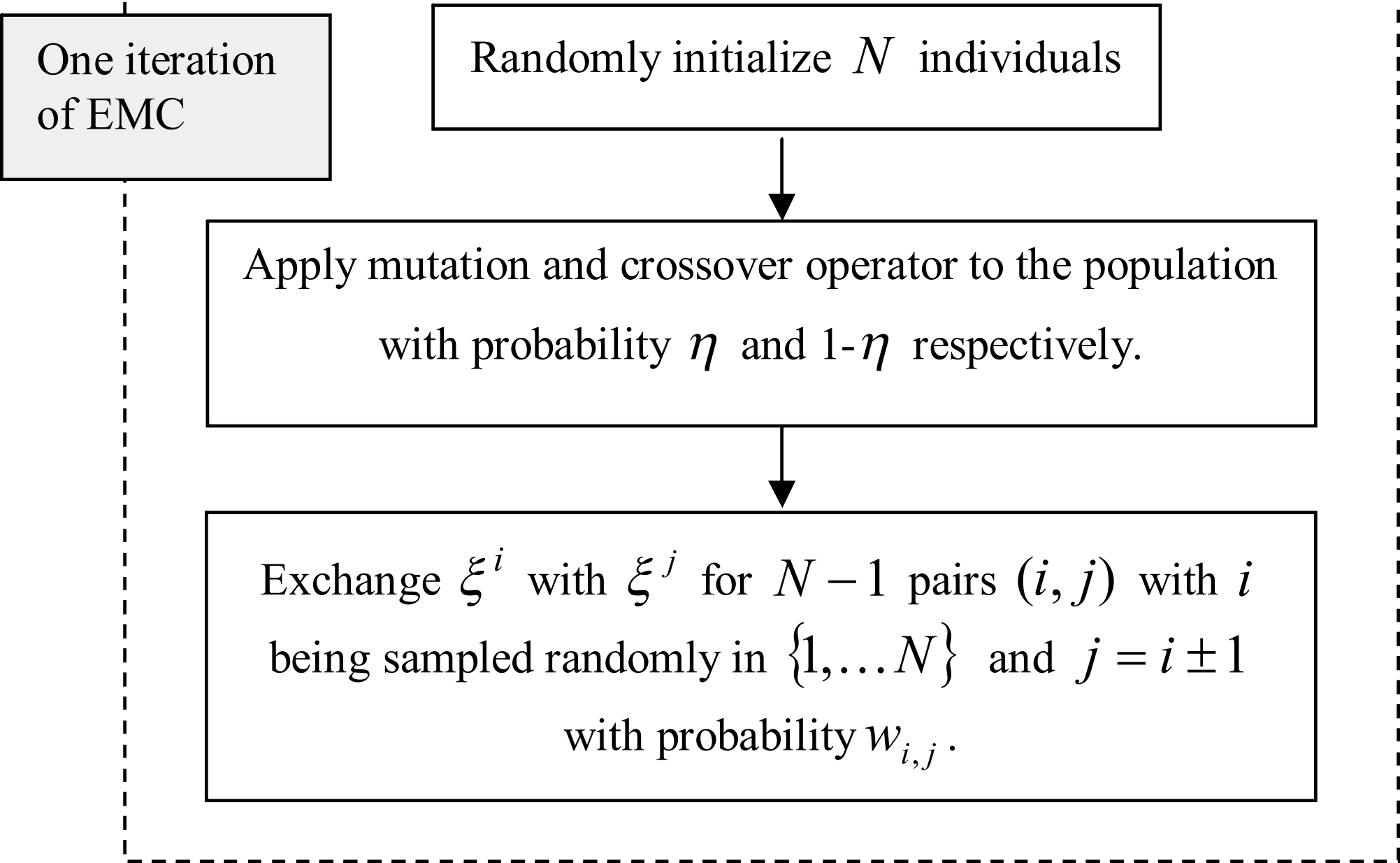
Figure 6. 95% modeling uncertainty intervals of streamflow simulation using different BNNs for days from May 28, 1975 to July 12, 1975 for the Reynolds Creek Experimental Watershed.

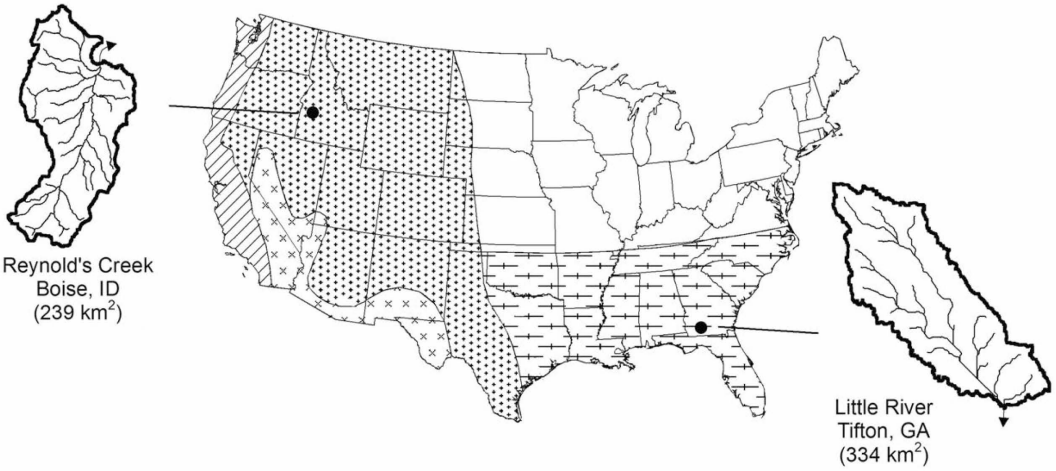
Figure 7. 95% predictive uncertainty intervals of streamflow simulation using different BNNs for days from May 28, 1972 to June 28, 1972 for the Reynolds Creek Experimental Watershed.

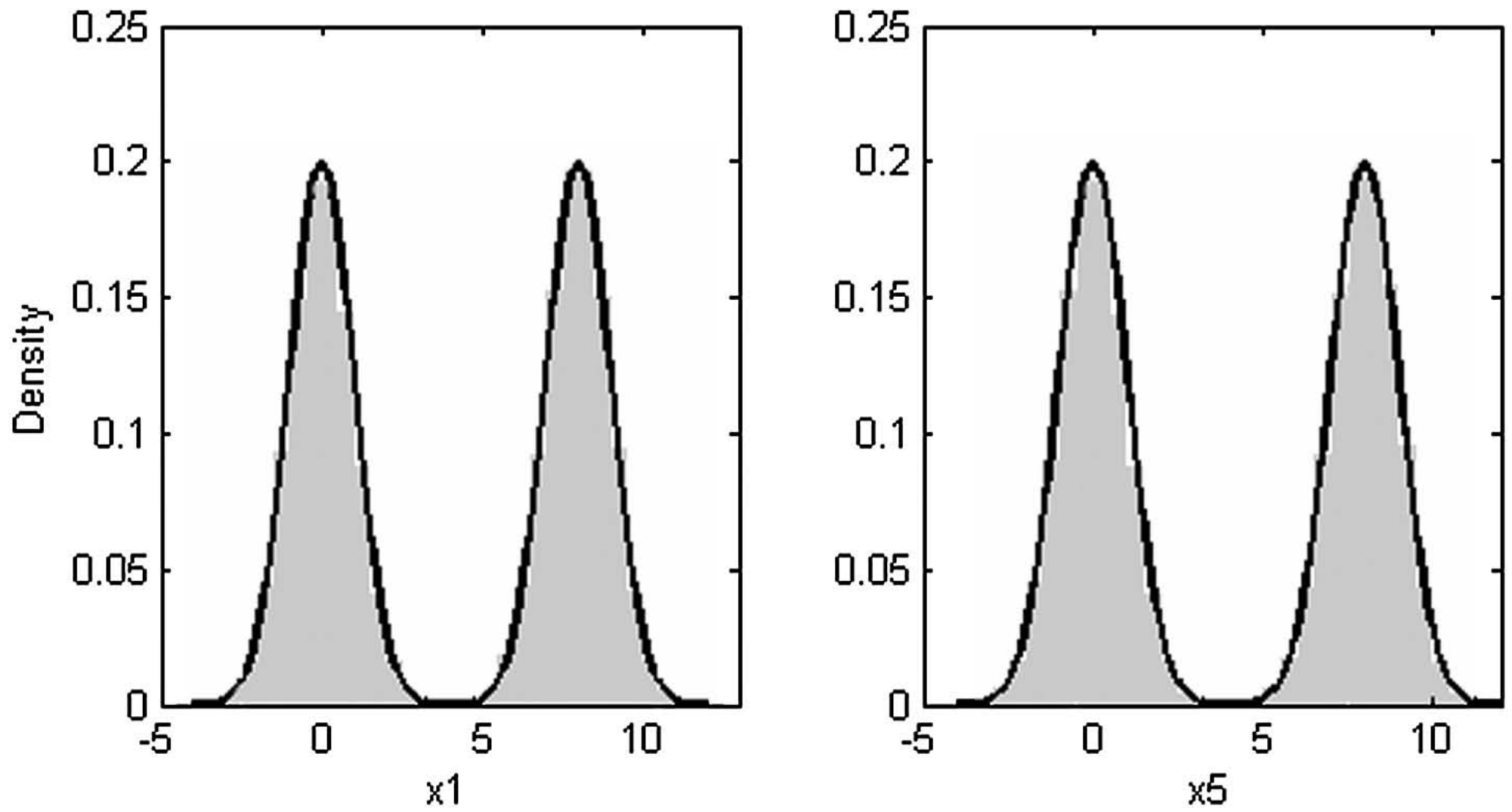
Figure 8. 95% modeling uncertainty intervals of streamflow simulation using different BNNs for days from January 4, 1997 to March 31, 1997 for the Little River Experimental Watershed.

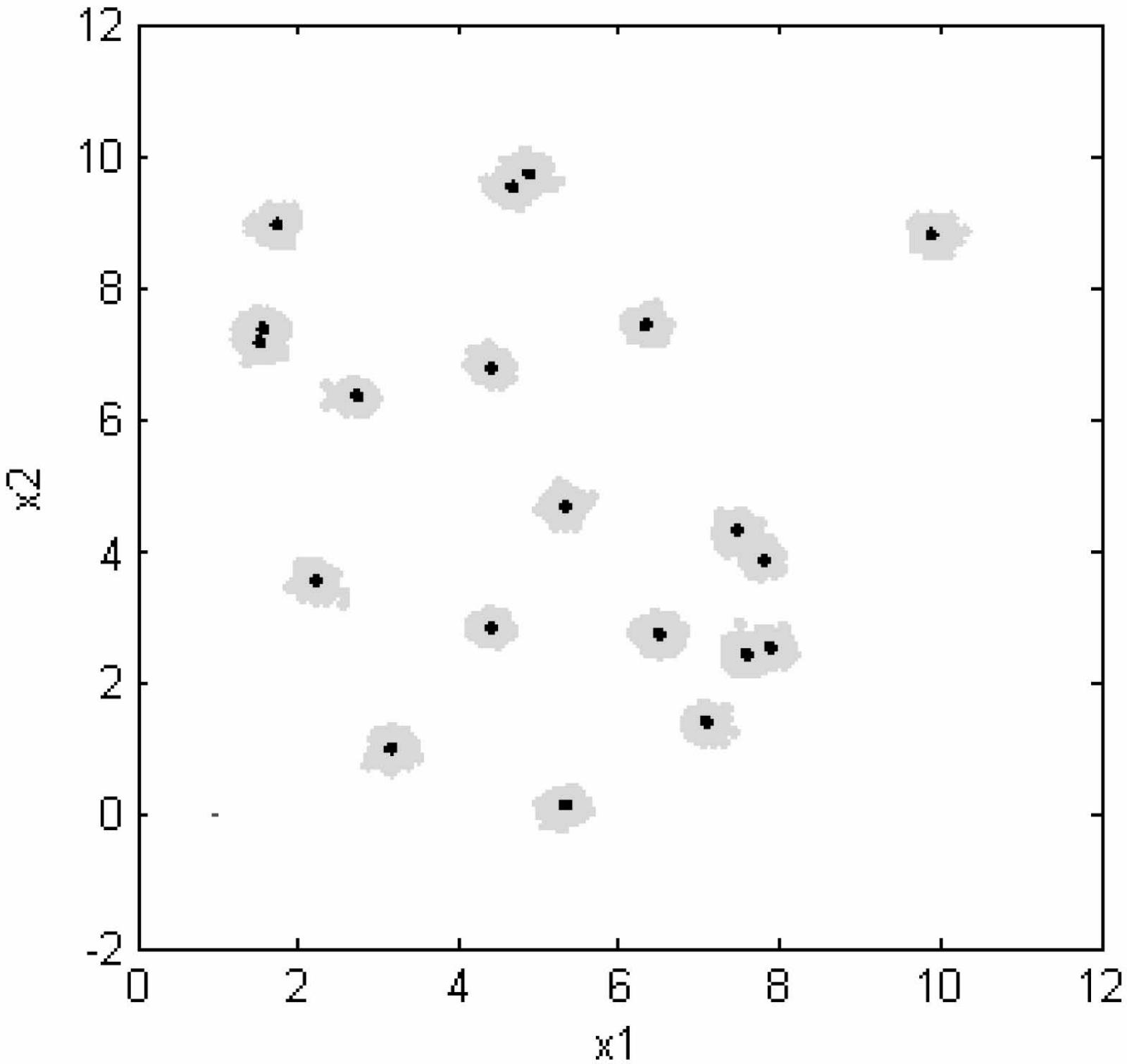
Figure 9. 95% predictive uncertainty intervals of streamflow simulation using different BNNs for days from January13, 2001 to April 24, 2001 for the Little River Experimental Watershed.

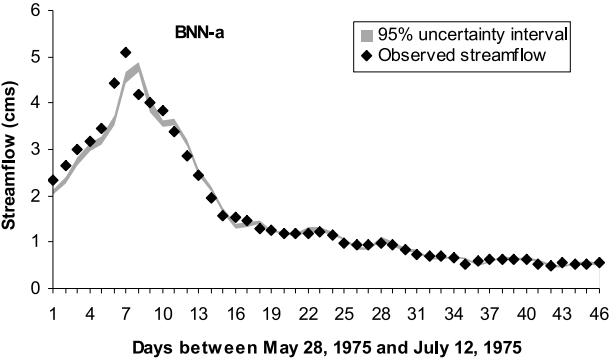

Bias
1
2
3
4
Input units
Hidden units
Output unit

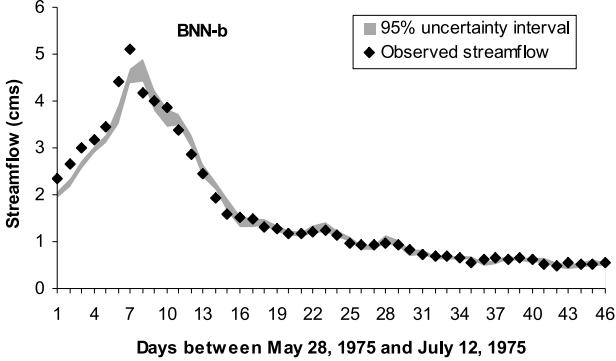

One iteration of EMC
Randomly initialize N individuals
Apply mutation and crossover operator to the population with probability η and 1-η respectively.
Exchange ξ^i with ξ^j for $N-1$ pairs (i, j) with i being sampled randomly in $\{1,\ldots N\}$ and $j = i \pm 1$ with probability $w_{i,j}$.

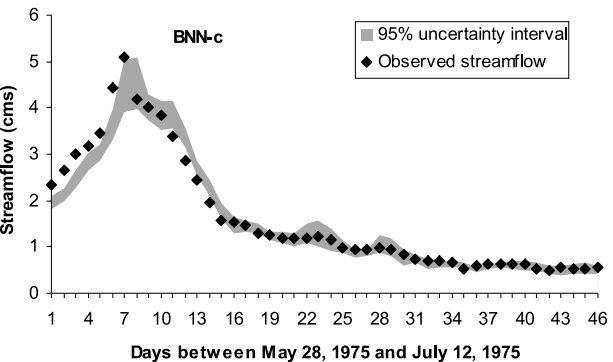

Reynold's Creek
Boise, ID
(239 km^2)
Little River
Tifton, GA
(334 km^2)

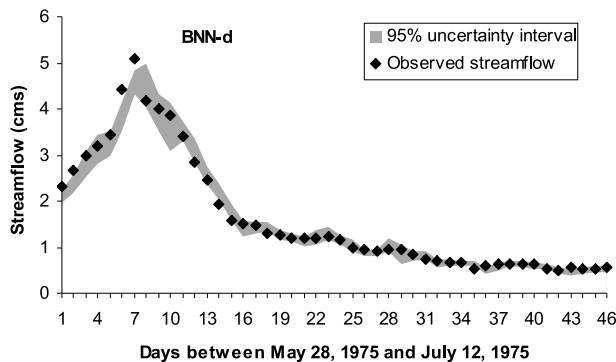

0.25
0.2
0.15
0.1
0.05
0
Density
-5
0
5
10
x1
0.25
0.2
0.15
0.1
0.05
0
-5
0
5
10
x5

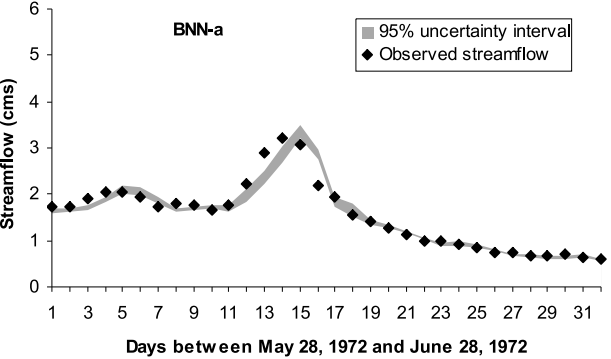

12
10
8
6
4
2
0
-2
x2
0
2
4
6
8
10
12
x1

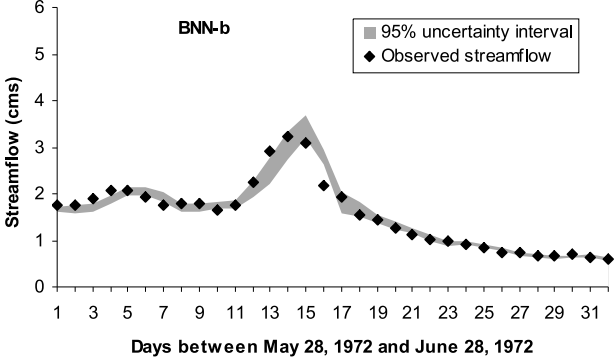

BNN-a
95% uncertainty interval
Observed streamflow
Streamflow (cms)
0
1
2
3
4
5
6
1 4 7 10 13 16 19 22 25 28 31 34 37 40 43 46
Days between May 28, 1975 and July 12, 1975

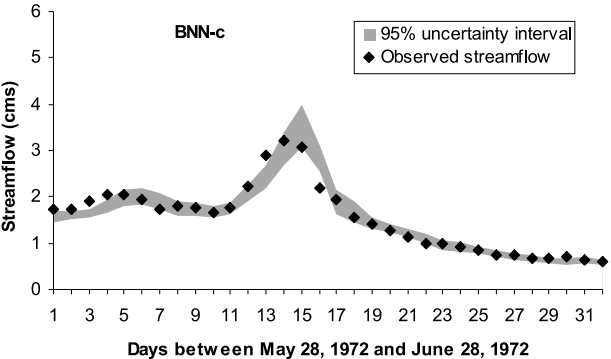

BNN-b
95% uncertainty interval
Observed streamflow
Streamflow (cms)
0
1
2
3
4
5
6
1 4 7 10 13 16 19 22 25 28 31 34 37 40 43 46
Days between May 28, 1975 and July 12, 1975

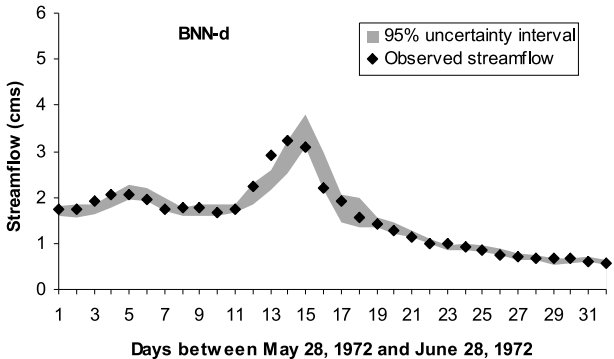

BNN-c
95% uncertainty interval
Observed streamflow
Streamflow (cms)
0
1
2
3
4
5
6
1 4 7 10 13 16 19 22 25 28 31 34 37 40 43 46
Days between May 28, 1975 and July 12, 1975

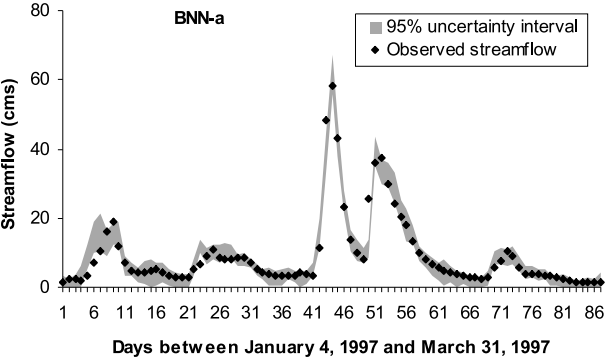

BNN-d
95% uncertainty interval
Observed streamflow
Streamflow (cms)
0
1
2
3
4
5
6
1 4 7 10 13 16 19 22 25 28 31 34 37 40 43 46
Days between May 28, 1975 and July 12, 1975

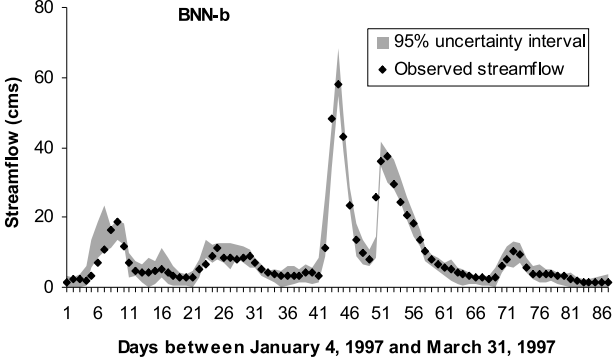

BNN-a
95% uncertainty interval
Observed streamflow
Streamflow (cms)
0
1
2
3
4
5
6
1 3 5 7 9 11 13 15 17 19 21 23 25 27 29 31
Days between May 28, 1972 and June 28, 1972

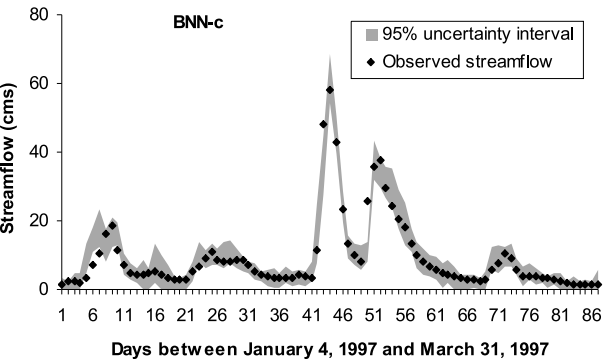

BNN-b
95% uncertainty interval
Observed streamflow
Streamflow (cms)
0
1
2
3
4
5
6
1 3 5 7 9 11 13 15 17 19 21 23 25 27 29 31
Days between May 28, 1972 and June 28, 1972

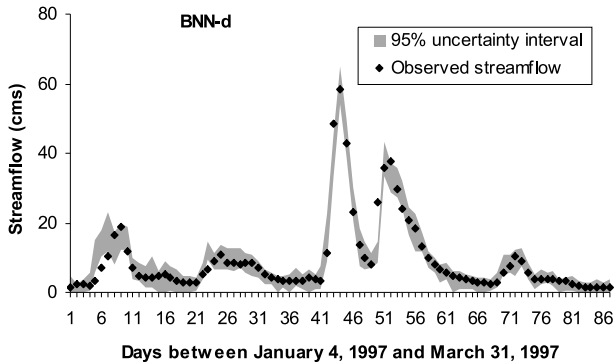

BNN-c
95% uncertainty interval
Observed streamflow
Streamflow (cms)
0
1
2
3
4
5
6
1 3 5 7 9 11 13 15 17 19 21 23 25 27 29 31
Days between May 28, 1972 and June 28, 1972

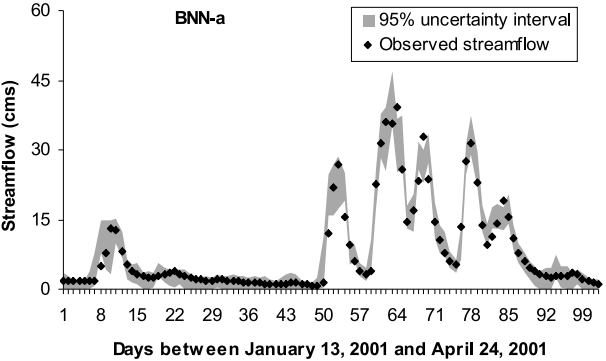

BNN-d
95% uncertainty interval
Observed streamflow
Streamflow (cms)
0
1
2
3
4
5
6
1 3 5 7 9 11 13 15 17 19 21 23 25 27 29 31
Days between May 28, 1972 and June 28, 1972

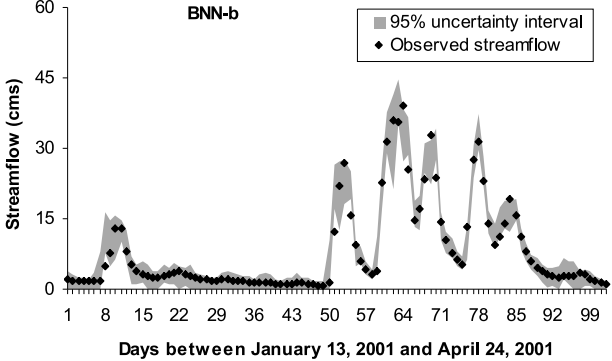

BNN-a
95% uncertainty interval
Observed streamflow
Streamflow (cms)
0
20
40
60
80
1 6 11 16 21 26 31 36 41 46 51 56 61 66 71 76 81 86
Days between January 4, 1997 and March 31, 1997

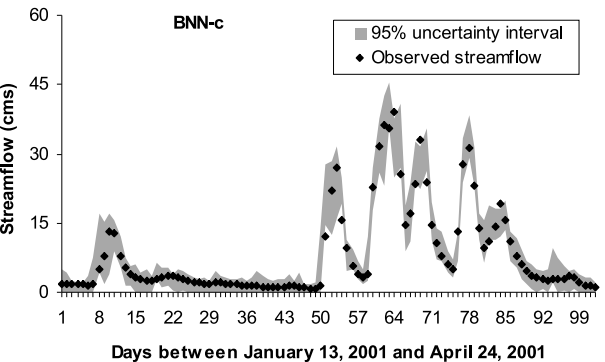

BNN-b
95% uncertainty interval
Observed streamflow
Streamflow (cms)
0
20
40
60
80
1 6 11 16 21 26 31 36 41 46 51 56 61 66 71 76 81 86
Days between January 4, 1997 and March 31, 1997

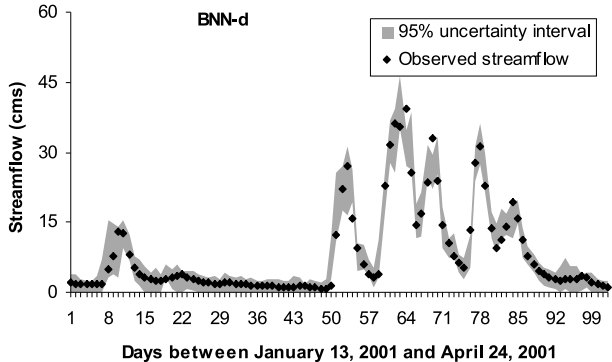

BNN-c
95% uncertainty interval
Observed streamflow
Streamflow (cms)
0
20
40
60
80
1 6 11 16 21 26 31 36 41 46 51 56 61 66 71 76 81 86
Days between January 4, 1997 and March 31, 1997


BNN-d
95% uncertainty interval
Observed streamflow
Streamflow (cms)
0
20
40
60
80
1 6 11 16 21 26 31 36 41 46 51 56 61 66 71 76 81 86
Days between January 4, 1997 and March 31, 1997


BNN-a
95% uncertainty interval
Observed streamflow
Streamflow (cms)
0
15
30
45
60
1
8
15
22
29
36
43
50
57
64
71
78
85
92
99
Days between January 13, 2001 and April 24, 2001


BNN-b
95% uncertainty interval
Observed streamflow
Streamflow (cms)
0
15
30
45
60
1
8
15
22
29
36
43
50
57
64
71
78
85
92
99
Days between January 13, 2001 and April 24, 2001


BNN-c
95% uncertainty interval
Observed streamflow
Streamflow (cms)
0
15
30
45
60
1
8
15
22
29
36
43
50
57
64
71
78
85
92
99
Days between January 13, 2001 and April 24, 2001


BNN-d
95% uncertainty interval
Observed streamflow
Streamflow (cms)
0
15
30
45
60
1
8
15
22
29
36
43
50
57
64
71
78
85
92
99
Days between January 13, 2001 and April 24, 2001