Spike-and-Slab Probabilistic Backpropagation: When Smarter Approximations Make No Difference

Evan Ott
Department of Statistics and Data Sciences
University of Texas at Austin
Austin, TX 78705
research@evanott.com

Sinead Williamson
Department of Statistics and Data Sciences
University of Texas at Austin
Austin, TX 78705
sinead@austin.utexas.edu

Abstract

Probabilistic backpropagation (PBP, Hernández-Lobato and Adams, 2015) is an approximate Bayesian inference method for deep neural networks, using a message-passing framework. These messages—which correspond to distributions arising as we propagate our input through a probabilistic neural network—are approximated as Gaussian. However, in practice, the exact distributions may be highly non-Gaussian. In this paper, we propose a more realistic approximation based on a spike-and-slab distribution. Unfortunately, in this case, better approximation of the messages does not translate to better downstream performance. We present results comparing the two schemes and discuss why we do not see a benefit from this spike-and-slab approach.

1 Introduction

Deep neural networks are flexible non-linear models common to domains with complicated data relationships. However, despite having many unknown model parameters, many neural networks do not attempt to represent model (epistemic) or data (aleatoric) uncertainty. Bayesian approaches to neural networks could capture this uncertainty but are typically rendered intractable in closed-form. MCMC-based methods (Neal, 1995; Cobb and Jalaian, 2021) offer asymptotic guarantees but are typically slow to converge. Instead, it is typical to use approximate inference algorithms, such as those based on Laplace approximations (MacKay, 1992; Daxberger et al., 2021) or variational inference (Graves, 2011; Blundell et al., 2015).

Probabilistic backpropagation (PBP, Hernández-Lobato and Adams, 2015) is an example of an approximation-based approach for Bayesian deep learning. Like many such approaches, it uses a mean-field approximation to the posterior, which is inferred via a message-passing algorithm based on assumed density filtering (Opper, 1999). The resulting algorithm is reminiscent of backpropagation; however, rather than propagating a single function estimate through a neural network, PBP propagates distributions representing our estimated posterior uncertainty. PBP approximates these distributions, or "messages," using a Gaussian, whose parameters are a known function of the means and variances of the per-weight distributions, allowing us to update those weight parameters using gradient information in a backwards pass.

In practice, the true messages can be highly non-Gaussian, since they have been propagated through a ReLU or similar nonlinearity. Using a Gaussian approximation risks ignoring sparsity inherent in the true message structure. In this paper, we show that the PBP algorithm can be modified to use sparse messages, parameterized using a spike-and-slab distribution. Computational costs are not significantly increased over Gaussian messages.

I Can't Believe It's Not Better Workshop at NeurIPS 2022.

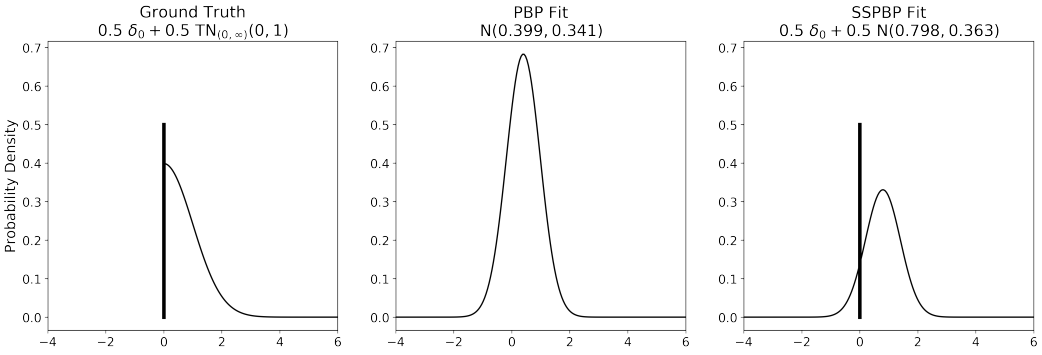

Figure 1: *Left*, the true distribution obtained by applying a ReLU to a $\mathrm{N}(0, 1)$ random variable, with a mixture of a truncated Gaussian and a "spike" (with its probability indicated with an overlayed vertical bar) at $X = 0$. *Center*, the Gaussian density obtained in the PBP approximation. *Right*, the spike-and-slab distribution obtained by our approximation (with the spike probability indicated with an overlayed vertical bar).

Unfortunately, we discover what many before us have discovered in different contexts: Gaussian approximations usually work pretty well. Closer investigation shows that, if we use bias terms in our neural network (as is typical), any theoretical advantage of our spike-and-slab approach evaporates. In the absence of a bias term, our approximation does differ from the standard PBP approach, and at a per-message level does indeed provide better approximations to the true message. However, this advantage does not translate into a significant difference in overall performance. While this is a negative result, and while it remains possible that alternative approximations could yield improved performance, our work indirectly highlights the fact that Gaussian approximations to non-Gaussian distributions are often a good choice in practice.

2 Probabilistic backpropagation

We consider the setting of feed-forward neural networks (FFNNs) with ReLU activation $\sigma(x) = \max(x, 0)$, such that

$$\begin{aligned} \mathbf{z}^{(\ell)}(\mathbf{x}) =& \sigma\left(\mathbf{W}^{(\ell)}\left[\mathbf{z}^{(\ell-1)}(\mathbf{x}); 1\right] \Big/ \sqrt{n_{\ell-1}+1}\right), \quad \ell = 1 : L-1 \\ \hat{\mathbf{y}} =& \mathbf{W}^{(L)}\left[\mathbf{z}^{(L-1)}(\mathbf{x}); 1\right] / \sqrt{n_{L-1}+1}\,, \end{aligned} \tag{1}$$

with $[a; b]$ indicating concatenation, n_ℓ indicating the number of nodes in layer ℓ, and $\mathbf{z}^{(0)}(\mathbf{x}) = \mathbf{x}$. Let $\mathcal{W} = (\mathbf{W}^{(\ell)})_{\ell=1}^{L}$ where $\mathbf{W}^{(\ell)} = (w_{ij}^{(\ell)})_{i=1}^{n_{\ell-1}+1}{}_{j=1}^{n_\ell}$ be the set of all weights (which includes the biases).[1] We include a scaling factor of $\sqrt{n_{\ell-1}+1}$ to make the scale of the input to each neuron invariant to the size of the previous layer, following Hernández-Lobato and Adams (2015).

To make the model Bayesian, we place a spherical Gaussian prior on the weights, meaning the $\mathbf{z}^{(\ell)}(\mathbf{x})$ and $\hat{\mathbf{y}}$ are now random, and we assume $y_i \sim \mathrm{N}(\hat{y}_i, \gamma^{-1})$. Since the number of weights can be large, and the ReLU nonlinearity removes the possibility of analytic tractability, posterior inference can be computationally challenging.

Probabilistic backpropagation (PBP, Hernández-Lobato and Adams, 2015) is an approximate inference technique appropriate for this setting. PBP uses assumed density filtering (ADF, Opper, 1999) to update a mean-field approximation to the true posterior,[2]

$$q(\mathcal{W}) = \prod_{\ell} q^{(\ell)}(\mathbf{W}^{(\ell)}) = \prod_{i,j,\ell} \mathrm{N}(w_{ij}^{(\ell)} | m_{ij}^{(\ell)}, v_{ij}^{(\ell)}). \tag{2}$$

[1] We adopt the convention throughout that x indicates a scalar; $\mathbf{x}$ a vector; $\mathbf{X}$ a matrix; and $\mathcal{X}$ a list of matrices.

[2] We ignore here terms concerning the observation noise variance and the prior variance of the Gaussians.

The mean and variance parameters of the weights ($m_{ij}^{(\ell)}$ and $v_{ij}^{(\ell)}$) are updated by propagating uncertainty through the network, that is, by evaluating the distributions over the representations $\mathbf{z}^{(\ell)}(\mathbf{x})$. PBP approximates these distributions with Gaussian distributions, obtained through moment matching. Leveraging some properties of Gaussian distributions (Minka, 2001), PBP is able to update the parameters of the approximate posterior in an approach analogous to backpropagation. Finally, PBP also incorporates an expectation propagation step to further refine the approximate posterior after each full pass through the training data.

Several extensions to the PBP framework have provided improvements, such as including approximations appropriate for classification (Ghosh et al., 2016), incorporating minibatching (Benatan and Pyzer-Knapp, 2018), and using non-diagonal Gaussians as the approximating distribution (Sun et al., 2017). Similar approaches have been used in a variational inference context (Roth and Pernkopf, 2016; Wu et al., 2019; Dera et al., 2019; Haußmann et al., 2020) and in a hybrid Bayesian/maximum likelihood context (Gast and Roth, 2018).

2.1 Limitations of a Gaussian approximation

PBP approximates the distribution of $\mathbf{z}^{(\ell)}(\mathbf{x})$—i.e., the "messages" in the ADF algorithm—using Gaussians. However, the ReLU nonlinearity in Equation 1 means that the true distributions can be highly non-Gaussian. As a simple example, consider the distribution over the jth element of the first-layer representation of an input $\mathbf{x}$, $z_j^{(1)}(\mathbf{x}) = \sigma({\mathbf{w}_j^{(1)}}^\top[\mathbf{x}; 1])$. We are approximating the posterior distributions over $w_{ij}^{(1)}$ using Gaussians, meaning that the distribution implied by $\mathbf{w}_j^\top[\mathbf{x}; 1]$ is also Gaussian. However, once this has passed through a ReLU, the distribution over $z_j^{(1)}(\mathbf{x})$ is a mixture of a Dirac delta distribution (or "spike") at $x = 0$ and a Gaussian truncated to the domain $(0, \infty)$. See Figure 1 for a demonstration.

3 Spike-and-slab probabilistic backpropagation

An exact implementation of ADF in the FFNN would send messages based on the true distribution over the $\mathbf{z}^{(\ell)}(\mathbf{x})$, given the current approximating distribution $q(\mathcal{W})$. We will refer to this true distribution as $q(\mathbf{z}^{(\ell)})$[3]. This is computationally infeasible: while we can calculate this distribution for a single layer where the input is the observed x (as shown above), on later layers the inputs are themselves the propagated distributions. PBP chooses to approximate these with diagonal Gaussian distributions $\widetilde{q}_{\text{PBP}}(\mathbf{z}^{(\ell)})$, obtained via moment matching.

We propose using a more sophisticated approximation of the messages $q(\mathbf{z}^{(\ell)})$. At the first layer, we know that $q(\mathbf{z}^{(1)})$ is a spike-and-slab distribution with truncated Gaussian slab. We choose to model the resulting sparsity directly using a spike-and-slab distribution with a (non-truncated) Gaussian slab, such that for node i in layer ℓ, we have

$$\widetilde{q}_{\text{SSPBP}}(z_i^{(\ell)}) = (1 - \widetilde{\rho}_i^{(\ell)})\delta_0(z_i^{(\ell)}) + \widetilde{\rho}_i^{(\ell)}\mathrm{N}(z_i^{(\ell)}|\widetilde{m}_i^{(\ell)}, \widetilde{v}_i^{(\ell)}), \tag{3}$$

with slab probability $\widetilde{\rho}_i^{(\ell)}$, slab mean $\widetilde{m}_i^{(\ell)}$, and slab variance $\widetilde{v}_i^{(\ell)}$. We refer to the resulting algorithm as Spike-and-Slab PBP (SSPBP).

3.1 Approximating messages using a spike-and-slab

We wish to approximate some distribution p, with a spike-and-slab distribution q of the form of Equation 3. We can do so by minimizing the Kullback-Leibler divergence $\mathrm{KL}(p\|q)$, yielding the following parameters (see Appendix A.1 for derivation):

$$\rho = \mathbb{P}_{X\sim p}[X \neq 0], \qquad m = \frac{1}{\rho}\mathbb{E}_{X\sim p}[X], \qquad v = \frac{1}{\rho}\left(\mathbb{V}_{X\sim p}[X] - \rho(1-\rho)m^2\right).$$

Note that this is similar to the moment-matching setting of PBP: we first match the spike probability $(1 - \rho)$, and then match the first and second moments.

[3]For notational conciseness, we hereafter write $\mathbf{z}^{(\ell)}$ in place of $\mathbf{z}^{(\ell)}(\mathbf{x})$

Table 1: Simulation study of how well SSPBP approximates the true distribution after a single layer, reporting the MMD between a ground truth sample and approximations obtained using either PBP or SSPBP, along with the MMD between two ground truth samples.

p_X	p_W	% Saturated	Same	PBP	SSPBP
$\mathrm{N}(0,1)$	$\mathrm{N}(0,1)$	49.86%	0.000028	0.066	**0.020**
$\mathrm{N}(1,1)$	$\mathrm{N}(3,1)$	16.24%	0.000090	0.031	**0.015**
$\mathrm{N}(1,1)$	$\mathrm{N}(-3,1)$	84.44%	0.000055	0.21	**0.0038**
$\mathrm{N}(3,1)$	$\mathrm{N}(3,1)$	0.22%	0.00025	**0.0043**	0.0051
$\mathrm{N}(3,1)$	$\mathrm{N}(-3,1)$	99.72%	0.00000058	0.017	**0.00024**

Consider the jth node of the ℓth layer of our FFNN. Following Hernández-Lobato and Adams (2015), we augment the incoming message with a bias term, so that $\widetilde{\rho}_{n_\ell+1}^{(\ell)} = 1$, $\widetilde{m}_{n_\ell+1}^{(\ell)} = 1$, and $\widetilde{v}_{n_\ell+1}^{(\ell)} = 0$. Let $\mathbf{M}^{(\ell)} = (\mathbf{m}_i^{(\ell)})_{i=1}^{n_\ell}$ and $\mathbf{m}_i^{(\ell)} = (m_{ij}^{(\ell)})_{j=1}^{n_{\ell-1}+1}$ be the means of the approximate posteriors $q(w_{ij}^{(\ell)})$ of the weights of the FFNN, and let $\mathbf{V}^{(\ell)} = (\mathbf{v}_i^{(\ell)})_{i=1}^{n_\ell}$ and $\mathbf{v}_i^{(\ell)} = (v_{ij}^{(\ell)})_{j=1}^{n_{\ell-1}+1}$ be the corresponding variances (Equation 2). In the absence of a nonlinearity (e.g., for a final regression layer), we approximate the message $z_j^{(\ell)}$ with a spike-and-slab distribution, with slab probability $\widetilde{\rho}_j^{(\ell,\text{linear})}$, slab mean $\widetilde{m}_j^{(\ell,\text{linear})}$, and slab variance $\widetilde{v}_j^{(\ell,\text{linear})}$ where

$$\widetilde{\rho}_j^{(\ell,\text{linear})} \equiv \widetilde{\rho}^{(\ell,\text{linear})} = 1 - \prod_{i=1}^{n_{\ell-1}+1} (1 - \widetilde{\rho}_i^{(\ell-1)}), \qquad \widetilde{m}_j^{(\ell,\text{linear})} = \frac{\mathbf{m}_j^{(\ell)\top}\left(\widetilde{\boldsymbol{\rho}}^{(\ell-1)} \circ \widetilde{\mathbf{m}}^{(\ell-1)}\right)}{\widetilde{\rho}^{(\ell,\text{linear})}\sqrt{n_{\ell-1}+1}},$$
$$\widetilde{v}_j^{(\ell,\text{linear})} = \frac{c_i}{(n_{\ell-1}+1)\,\widetilde{\rho}^{(\ell,\text{linear})}} - \left(1 - \widetilde{\rho}^{(\ell,\text{linear})}\right)\left(\widetilde{m}_j^{(\ell,\text{linear})}\right)^2 \tag{4}$$

where $\circ$ indicates the element-wise Hadamard product and

$$\begin{aligned}\mathbf{c} =& \mathbf{V}^{(\ell)}\left(\widetilde{\boldsymbol{\rho}}^{(\ell-1)} \circ \widetilde{\mathbf{m}}^{(\ell-1)} \circ \widetilde{\mathbf{m}}^{(\ell-1)}\right) + \left(\mathbf{M}^{(\ell)} \circ \mathbf{M}^{(\ell)}\right)\left(\widetilde{\boldsymbol{\rho}}^{(\ell-1)} \circ \widetilde{\mathbf{v}}^{(\ell-1)}\right) \\ &+ \mathbf{V}^{(\ell)}\left(\widetilde{\boldsymbol{\rho}}^{(\ell-1)} \circ \widetilde{\mathbf{v}}^{(\ell-1)}\right) + \left(\mathbf{M}^{(\ell)} \circ \mathbf{M}^{(\ell)}\right)\left(\widetilde{\boldsymbol{\rho}}^{(\ell-1)} \circ \left(1 - \widetilde{\boldsymbol{\rho}}^{(\ell-1)}\right) \circ \widetilde{\mathbf{m}}^{(\ell-1)} \circ \widetilde{\mathbf{m}}^{(\ell-1)}\right).\end{aligned}$$

If we pass this through a ReLU, we have

$$\widetilde{\rho}_j^{(\ell,\text{ReLU})} = \widetilde{\rho}^{(\ell,\text{linear})}\Phi\left(\alpha_j\right), \qquad \widetilde{m}_j^{(\ell,\text{ReLU})} = \widetilde{m}_j^{(\ell,\text{linear})} + \gamma_j\sqrt{\widetilde{v}_j^{(\ell,\text{linear})}},$$
$$\widetilde{v}_j^{(\ell,\text{ReLU})} = \left(1 - \gamma_j\alpha_j - \gamma_j^2\right)\widetilde{v}_j^{(\ell,\text{linear})}$$

with intermediate values $\alpha_j = \widetilde{m}_j^{(\ell,\text{linear})}/\sqrt{\widetilde{v}_j^{(\ell,\text{linear})}}$ and $\gamma_j = \phi\left(\alpha_j\right)/\Phi\left(\alpha_j\right)$, where ϕ and Φ are the PDF and CDF of a standard Gaussian, respectively. Derivations are provided in Appendix A.

4 Empirical analysis

We begin by numerically validating our approximation before exploring how well it performs when used in a Bayesian FFNN. Code can be found at the anonymous repository `https://github.com/SSPBP/SSPBPcode`. All hyperparameters follow those used by Hernández-Lobato and Adams (2015), unless otherwise stated.

4.1 Quality of the spike-and-slab approximation

To explore why the spike-and-slab approximation should yield a better representation of the propagated probability distributions than a Gaussian, we performed a simulation study. Here, we explore a ReLU transformation $T = \text{ReLU}(XW)$ applied to the product of two independent Gaussians X and W. We apply the PBP and SSPBP approximations to determine how the two approaches would approximate the resultant distribution, then we compare 10,000 samples of the approximate

distributions to a ground truth sample by computing the maximum mean discrepancy (MMD) with a squared exponential kernel $K(x, y) = \exp(-\gamma(x-y)^2)$, setting $\gamma = 1$. The results are summarized in Table 1. The first two columns show the parameters of the underlying distributions, and the third column shows the percent of the distribution that is saturated to zero by the ReLU nonlinearity. The fourth column shows the MMD between two size-10,000 samples from the true distribution, to give an idea of the scale of the differences. Next, we see the MMD between the PBP approximation and the true distribution (column 5) and the MMD between our SSPBP approximation and the true distribution (column 6). We see from these results that, in cases where the true distribution has non-trivial sparsity, samples from SSPBP more closely match the true distribution compared with PBP, giving credence to the intuition that a spike-and-slab approximation should improve the PBP framework.

4.2 Evaluation as part of a Bayesian FFNN

Table 2: Mean and standard error of average test set RMSE of PBP and SSPBP, on eight datasets.

	1 Layer 50 Nodes		**2 Layers 10 Nodes**	
Dataset	**PBP**	**SSPBP**	**PBP**	**SSPBP**
Boston Housing	**3.554±0.179**	3.566±0.198	3.097±0.147	**2.997±0.165**
Combined Cycle Power Plant	4.117±0.037	**4.116±0.033**	**4.088±0.067**	4.096±0.066
Concrete Compression Strength	**5.616±0.125**	5.662±0.105	6.031±0.161	**5.921±0.158**
Energy Efficiency	1.857±0.081	**1.856±0.079**	**1.477±0.043**	1.660±0.112
Kin8nm	**0.098±0.001**	0.100±0.001	0.111±0.004	**0.109±0.002**
Naval Propulsion	**0.006±0.000**	**0.006±0.000**	**0.006±0.000**	**0.006±0.000**
Wine Quality Red	0.655±0.003	**0.654±0.003**	0.653±0.012	**0.652±0.008**
Yacht Hydrodynamics	**1.344±0.061**	1.367±0.071	**1.064±0.072**	1.131±0.063

Since our spike-and-slab approximation mimics the sparsity inherent in the FFNN, we expected it to achieve better performance. To evaluate this, we compared our approach to PBP, modifying the existing official Theano (Theano Development Team, 2016) implementation, which was released under a BSD 3-Clause "New" or "Revised" License.[4] We applied several different model architectures to eight regression datasets used in Hernández-Lobato and Adams (2015). We trained the models using 40 training epochs (without early stopping), with an 80%-20% training-test split, repeating the process five times for each dataset-model comparison (following the original PBP settings). We report the mean and standard error of the average test set RMSE across those five repetitions in Table 2. Running all experiments in Tables 2 and 3 took under 12 hours on an Intel i5 4 core 3.2 GHz desktop, using CPU computation.

As Table 2 indicates, the performance is comparable across the two models. Disappointingly, there is no significant difference between the two approaches.

A hint at why this is can be seen by considering the estimate of the spike probability in Equation 4. Our approximation *only* captures spikes at zero. However, if our FFNN includes bias terms, this bias will translate spikes at zero in $\mathbf{z}^{(\ell-1)}$ to non-zero locations in $\mathbf{z}^{(\ell)}$, meaning our approximation will not capture them. This means all mass is placed on the slab, rendering our approximation *identical* to modeling solely with a Gaussian as in the original PBP framework. We show this formally in Appendix B. The variation in Table 2 is likely due to numerical instability, using different random seeds, or other similar issues, not as a result of improving the internal approximations of the PBP framework.

The algorithms *do* however differ if we do not include a bias term. In Table 3 we repeat our experiments without a bias term. Unfortunately, despite the fact that we are now indeed comparing different update rules, there remains no significant difference between the algorithms.

We hypothesise that this may be because the slab probabilities $\widetilde{\rho}^{(\ell,\text{linear})}$ would tend toward one given that the product in Equation 4 will tend toward zero in all but the most extreme cases. To test this,

[4] `https://github.com/HIPS/Probabilistic-Backpropagation`

Table 3: Mean and standard error of average test set RMSE for the bias-free versions of PBP and SSPBP, on eight datasets.
† Due to numerical issues, the trials for this model were repeated with a different random seed.

	1 Layer 50 Nodes		**2 Layers 10 Nodes**	
Dataset	**PBP**	**SSPBP**	**PBP**	**SSPBP**
Boston Housing	**3.529±0.296**	3.544±0.301	**3.809±0.295**	3.865±0.322
Combined Cycle Power Plant	4.300±0.054	**4.295±0.052**	4.190±0.017	**4.188±0.023**
Concrete Compression Strength	7.092±0.108	**7.010±0.156**	6.823±0.214	**6.668±0.237**
Energy Efficiency	**1.788±0.038**	1.789±0.039	1.699±0.041	**1.617±0.019**
Kin8nm	0.159±0.002	**0.158±0.002**	0.126±0.001	**0.125±0.001**†
Naval Propulsion	**0.006±0.000**	**0.006±0.000**	**0.005±0.000**	0.006±0.000
Wine Quality Red	0.621±0.013	**0.618±0.013**	0.635±0.011	**0.633±0.014**
Yacht Hydrodynamics	**6.200±0.269**	6.319±0.282	**3.898±0.245**	4.276±0.390

we compute $\widetilde{\rho}^{(\ell,\text{linear})}$ for the linear layers in several architectures trained on the Boston dataset. See Table 4 for results. Note that $\widetilde{\rho}^{(1,\text{linear})}$ is always one by construction, since the signal has not passed through any nonlinearities. For later layers, even though the signal *has* passed through nonlinearities, we see that the slab probability does tend towards one in practice, even for narrow networks.

Table 4: Mean and standard error of the average slab probability in linear layers, $\widetilde{\rho}^{(\ell,\text{linear})}$, on test set observations for various architectures on the Boston dataset. 30 trials were completed for each choice of hidden layers.
† Some trials removed due to numerical instabilities.

Hidden Layers	$\widetilde{\rho}^{(1,\text{linear})}$	$\widetilde{\rho}^{(2,\text{linear})}$	$\widetilde{\rho}^{(3,\text{linear})}$	Output $\hat{y}$
5	1.000 ± 0.000	–	–	0.976 ± 0.007
50	1.000 ± 0.000	–	–	1.000 ± 0.000
5×5	1.000 ± 0.000	0.962 ± 0.007	–	$0.968 \pm 0.006^{\dagger}$
50×50	1.000 ± 0.000	1.000 ± 0.000	–	1.000 ± 0.000
$5 \times 5 \times 5$	1.000 ± 0.000	0.958 ± 0.008	$0.970 \pm 0.008^{\dagger}$	$0.971 \pm 0.009^{\dagger}$
$50 \times 50 \times 50$	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000

5 Discussion

In this paper, we presented SSPBP, a spike-and-slab variant of probabilistic backpropagation. Ultimately, we determine that, empirically and analytically, the use of a spike-and-slab approximation does not improve performance, despite seeming to be a more intuitive approximation for the problem. While using a spike-and-slab approximate posterior distribution for model parameters could—in theory—provide better results in a bias-free FFNN, our investigation of this setting casts doubt on this intuition: the spike-and-slab approximation *is* able to model sparsity inherent to the ReLU activation function but fails to produce better empirical results, likely due to the fact that, in practice, the slab probability tends to saturate towards one. Possible future directions could include assessing alternative approximations or incorporating spike-and-slab approximations in formulations of approximate Bayesian inference like variational inference; however, it appears that a Gaussian approximation is hard to beat.

References

Matt Benatan and Edward O Pyzer-Knapp. Practical considerations for probabilistic backpropagation. In *NeurIPS Workshop on Bayesian Deep Learning*, 2018.

Charles Blundell, Julien Cornebise, Koray Kavukcuoglu, and Daan Wierstra. Weight uncertainty in neural networks. In *International Conference on Machine Learning*, 2015.

Adam D Cobb and Brian Jalaian. Scaling Hamiltonian Monte Carlo inference for Bayesian neural networks with symmetric splitting. In *Uncertainty in Artificial Intelligence*, 2021.

Erik Daxberger, Agustinus Kristiadi, Alexander Immer, Runa Eschenhagen, Matthias Bauer, and Philipp Hennig. Laplace redux-effortless Bayesian deep learning. *Advances in Neural Information Processing Systems*, 2021.

Dimah Dera, Ghulam Rasool, and Nidhal Bouaynaya. Extended variational inference for propagating uncertainty in convolutional neural networks. In *IEEE International Workshop on Machine Learning for Signal Processing*, 2019.

Jochen Gast and Stefan Roth. Lightweight probabilistic deep networks. In *Computer Vision and Pattern Recognition*, 2018.

Soumya Ghosh, Francesco Maria Delle Fave, and Jonathan Yedidia. Assumed density filtering methods for learning Bayesian neural networks. In *AAAI Conference on Artificial Intelligence*, 2016.

Alex Graves. Practical variational inference for neural networks. *Advances in Neural Information Processing Systems*, 24, 2011.

Manuel Haußmann, Fred A Hamprecht, and Melih Kandemir. Sampling-free variational inference of Bayesian neural networks by variance backpropagation. In *Uncertainty in Artificial Intelligence*, 2020.

José Miguel Hernández-Lobato and Ryan Adams. Probabilistic backpropagation for scalable learning of Bayesian neural networks. In *International Conference on Machine Learning*, 2015.

David JC MacKay. Bayesian interpolation. *Neural computation*, 4(3):415–447, 1992.

Thomas Peter Minka. *A family of algorithms for approximate Bayesian inference*. PhD thesis, Massachusetts Institute of Technology, 2001.

Radford M Neal. *Bayesian learning for neural networks*. PhD thesis, University of Toronto, 1995.

Manfred Opper. *On-line learning in neural networks*, chapter A Bayesian approach to on-line learning, page 363–378. Publications of the Newton Institute. Cambridge University Press, 1999.

Wolfgang Roth and Franz Pernkopf. Variational inference in neural networks using an approximate closed-form objective. In *NeurIPS Workshop on Bayesian Deep Learning*, 2016.

Shengyang Sun, Changyou Chen, and Lawrence Carin. Learning structured weight uncertainty in Bayesian neural networks. In *International Conference on Artificial Intelligence and Statistics*, 2017.

Theano Development Team. Theano: A Python framework for fast computation of mathematical expressions. *arXiv e-prints*, abs/1605.02688, May 2016.

Anqi Wu, Sebastian Nowozin, Edward Meeds, Richard E. Turner, Jose Miguel Hernandez-Lobato, and Alexander L. Gaunt. Deterministic variational inference for robust Bayesian neural networks. In *International Conference on Learning Representations*, 2019.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes]
   (c) Did you discuss any potential negative societal impacts of your work? [No] Since the paper explores unsuccessful alternatives to an existing algorithm, we do not believe there to be any negative societal impacts.

(d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]

2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [N/A]
   (b) Did you include complete proofs of all theoretical results? [N/A]
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] We include a link to an anonymous GitHub repository.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] Unless otherwise stated, we used the same parameter settings as Hernández-Lobato and Adams (2015).
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes]
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes]
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes]
   (b) Did you mention the license of the assets? [Yes] We included license information where available.
   (c) Did you include any new assets either in the supplemental material or as a URL? [Yes]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [No] All datasets are commonly used and publicly available and do not include human-level information.
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [No] All datasets are commonly used and publicly available and do not include human-level information, or text/images.
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]

A Approximating messages using a spike-and-slab distribution

We take the following for our approximating distribution q, that is, the spike and (non-central) slab:

$$\widetilde{q}(z; \widetilde{\rho}, \widetilde{m}, \widetilde{v}) = (1 - \widetilde{\rho})\delta_0(z) + \widetilde{\rho}\mathrm{N}(z; \widetilde{m}, \widetilde{v}).$$

We seek to minimize $KL(p\|\widetilde{q})$ with respect to $\widetilde{\rho}, \widetilde{m}, \widetilde{v}$. Were $\widetilde{q}$ a Gaussian, this would be solved by moment matching $\widetilde{m} = \mathbb{E}_p[X]$ and $\widetilde{v} = \mathbb{V}_p[X]$ (Minka, 2001). Here, we derive the appropriate values of $\widetilde{\rho}, \widetilde{m}, \widetilde{v}$. For simplicity, and since we make use of a mean-field approximation, we'll focus on a univariate version, similar to PBP's approach:

$$\min_{\widetilde{q}} KL(p\|\widetilde{q}) \propto -\int_{\mathbb{R}} p(z)\log(\widetilde{q}(z))dz = -\int_{\mathbb{R}} p(z)\log\left((1-\widetilde{\rho})\delta_0(z) + \widetilde{\rho}\mathrm{N}(z;\widetilde{m},\widetilde{v})\right)dz.$$

We begin by giving values for $\widetilde{m}$, $\widetilde{v}$ and $\widetilde{\rho}$ in terms of the mean, variance, and probability at zero of the distribution being approximated.

A.1 Approximating a distribution with known mean, variance and probability of zero

Slab mean parameter $\widetilde{m}$ We begin by seeking to minimize $KL(p\|\widetilde{q})$ with respect to the slab's mean parameter, $\widetilde{m}$:

$$
\begin{aligned}
-\frac{d}{d\widetilde{m}}\int_{\mathbb{R}} p(z)\log(\widetilde{q}(z))dz &= -\int_{\mathbb{R}} p(z)\frac{\frac{\partial}{\partial m}\widetilde{q}(z;\widetilde{\rho},\widetilde{m},\widetilde{v})}{\widetilde{q}(z;\rho,\widetilde{m},\widetilde{v})}dz \\
&= -\int_{\mathbb{R}} p(z)\frac{\frac{\partial}{\partial\widetilde{m}}\left((1-\widetilde{\rho})\delta_0(z)+\widetilde{\rho}\mathrm{N}(z;\widetilde{m},\widetilde{v})\right)}{(1-\widetilde{\rho})\delta_0(z)+\widetilde{\rho}\mathrm{N}(z;\widetilde{m},\widetilde{v})}dz \\
&= -\int_{\mathbb{R}} p(z)\frac{\widetilde{\rho}\mathrm{N}(z;\widetilde{m},\widetilde{v})\left(\frac{z-\widetilde{m}}{\widetilde{v}}\right)+0}{(1-\widetilde{\rho})\delta_0(z)+\widetilde{\rho}\mathrm{N}(z;\widetilde{m},\widetilde{v})}dz \\
&= -\int_{\mathbb{R}} p(z)\frac{\widetilde{\rho}\mathrm{N}(z;\widetilde{m},\widetilde{v})\left(\frac{z-\widetilde{m}}{\widetilde{v}}\right)+(1-\widetilde{\rho})\delta_0(z)\left(\frac{z-\widetilde{m}}{\widetilde{v}}\right)-(1-\widetilde{\rho})\delta_0(z)\left(\frac{z-\widetilde{m}}{\widetilde{v}}\right)}{(1-\widetilde{\rho})\delta_0(z)+\widetilde{\rho}\mathrm{N}(z;\widetilde{m},\widetilde{v})}dz \\
&= -\int_{\mathbb{R}} p(z)\left(\frac{z-\widetilde{m}}{\widetilde{v}}\right)dz+\int_{\mathbb{R}} p(z)\frac{(1-\widetilde{\rho})\delta_0(z)\left(\frac{z-\widetilde{m}}{\widetilde{v}}\right)}{(1-\widetilde{\rho})\delta_0(z)+\widetilde{\rho}\mathrm{N}(z;\widetilde{m},\widetilde{v})}dz \\
&= -\mathbb{E}_p\left[\frac{Z-\widetilde{m}}{\widetilde{v}}\right]+\int_{\mathbb{R}} p(z)\frac{(1-\widetilde{\rho})\delta_0(z)\left(\frac{z-\widetilde{m}}{\widetilde{v}}\right)}{(1-\widetilde{\rho})\delta_0(z)+\widetilde{\rho}\mathrm{N}(z;\widetilde{m},\widetilde{v})}dz.
\end{aligned}
\tag{5}
$$

The integral in the final line of Equation 5 is nontrivial. We approximate this term by considering the delta function as a limit of the uniform distribution $u_a(z) = \mathrm{Unif}(z;[-1/2a,1/2a])$, such that as $a\to\infty$, we recover the delta function at 0 $\delta_0(z)$. Because u_a is 0 outside the range $[-1/2a,1/2a]$, we can restrict the domain of the integral:

$$
\begin{aligned}
I &= \lim_{a\to\infty}\int_{\mathbb{R}} p(z)\frac{(1-\widetilde{\rho})u_a(z)}{(1-\widetilde{\rho})u_a(z)+\rho\mathrm{N}(z;\widetilde{m},\widetilde{v})}f(z)dz \\
&= \lim_{a\to\infty}\int_{-1/2a}^{1/2a} p(z)\frac{(1-\widetilde{\rho})a}{(1-\widetilde{\rho})a+\widetilde{\rho}\mathrm{N}(z;\widetilde{m},\widetilde{v})}f(z)dz.
\end{aligned}
$$

Under the assumption that $(1-\widetilde{\rho})a \gg \rho\mathrm{N}(z;\widetilde{m},\widetilde{v})$ (as $a\to\infty$), we have

$$
I \approx \lim_{a\to\infty}\int_{-1/2a}^{1/2a} p(z)\frac{(1-\widetilde{\rho})a}{(1-\widetilde{\rho})a}f(z)dz = \lim_{\epsilon\to 0^+}\int_{-\epsilon}^{\epsilon} p(z)f(z)dz.
$$

Setting the derivative in Equation 5 to zero, we have

$$
\begin{aligned}
\mathbb{E}_p[Z] &\approx \widetilde{m}+\lim_{\epsilon\to 0^+}\int_{-\epsilon}^{\epsilon} p(z)\,(z-\widetilde{m})\,dz \\
&= \widetilde{m}-\widetilde{m}\lim_{\epsilon\to 0^+}\int_{-\epsilon}^{\epsilon} p(z)dz+\lim_{\epsilon\to 0^+}\int_{-\epsilon}^{\epsilon} p(z)zdz \\
&= \widetilde{m}-\widetilde{m}\mathbb{P}_p\left[Z=0\right]+0.
\end{aligned}
$$

This yields our solution

$$
\widetilde{m} = \frac{\mathbb{E}_p[Z]}{1-\mathbb{P}_p\left[Z=0\right]}.
$$

Slab variance parameter $\widetilde{v}$ For the variance parameter $\widetilde{v}$, we similarly obtain:

$$\begin{aligned}
\frac{\partial \widetilde{q}(z;\widetilde{\rho},\widetilde{m},\widetilde{v})}{\partial \widetilde{v}} &= \widetilde{\rho}\frac{1}{\sqrt{2\pi\widetilde{v}}}\exp\left(-\frac{(z-\widetilde{m})^2}{2\widetilde{v}}\right)\left(\frac{(z-\widetilde{m})^2-\widetilde{v}}{2\widetilde{v}^2}\right) \\
&= \widetilde{\rho}\mathbf{N}(z;\widetilde{m},\widetilde{v})\left(\frac{(z-\widetilde{m})^2-\widetilde{v}}{2\widetilde{v}^2}\right) \\
0 = -\frac{d}{d\widetilde{v}}\int_{\mathbb{R}} p(z)\log(\widetilde{q}(z))dz &= -\int_{\mathbb{R}} p(z)\frac{\frac{\partial}{\partial\widetilde{v}}\widetilde{q}(z;\widetilde{\rho},\widetilde{m},\widetilde{v})}{\widetilde{q}(z;\widetilde{\rho},\widetilde{m},\widetilde{v})}dz \\
&= -\int_{\mathbb{R}} p(z)\frac{\widetilde{\rho}\mathbf{N}(z;\widetilde{m},\widetilde{v})\left(\frac{(z-\widetilde{m})^2-\widetilde{v}}{2\widetilde{v}^2}\right)}{(1-\widetilde{\rho})\delta_0(z)+\widetilde{\rho}\mathbf{N}(z;\widetilde{m},\widetilde{v})}dz \\
&= -\mathbb{E}_p\left[\frac{(Z-\widetilde{m})^2-\widetilde{v}}{2\widetilde{v}^2}\right]+\int_{\mathbb{R}} p(z)\frac{(1-\widetilde{\rho})\delta_0(z)\left(\frac{(z-\widetilde{m})^2-\widetilde{v}}{2\widetilde{v}^2}\right)}{(1-\widetilde{\rho})\delta_0(z)+\widetilde{\rho}\mathbf{N}(z;\widetilde{m},\widetilde{v})}dz \\
&\approx -\mathbb{E}_p\left[\frac{(Z-\widetilde{m})^2-\widetilde{v}}{2\widetilde{v}^2}\right]+\lim_{\epsilon\to 0^+}\int_{-\epsilon}^{\epsilon} p(z)\left(\frac{(z-\widetilde{m})^2-\widetilde{v}}{2\widetilde{v}^2}\right)dz.
\end{aligned}$$

Thus, we have

$$\begin{aligned}
\mathbb{E}_p[Z^2]-2\widetilde{m}\mathbb{E}_p[Z]+\widetilde{m}^2 &\approx \widetilde{v}\left(1-\lim_{\epsilon\to 0^+}\int_{-\epsilon}^{\epsilon} p(z)dz\right) \\
&\quad+\lim_{\epsilon\to 0^+}\int_{-\epsilon}^{\epsilon} p(z)(z^2-2z\widetilde{m}+\widetilde{m}^2)dz \\
&= \widetilde{v}(1-\mathbb{P}_p[Z=0])+\widetilde{m}^2\mathbb{P}_p[Z=0]+0.
\end{aligned}$$

This yields

$$\begin{aligned}
\widetilde{v} &= \frac{\mathbb{E}_p[Z^2]-2\widetilde{m}\mathbb{E}_p[Z]+\widetilde{m}^2(1-\mathbb{P}_p[Z=0])}{1-\mathbb{P}_p[Z=0]} \\
&= \frac{1}{1-\mathbb{P}_p[Z=0]}\left(\mathbb{V}_p[Z]-\frac{\mathbb{E}_p[Z]^2\mathbb{P}_p[Z=0]}{1-\mathbb{P}_p[Z=0]}\right).
\end{aligned}$$

Slab probability parameter $\widetilde{\rho}$ Finally, for the mass of the slab $\widetilde{\rho}$, we have

$$\begin{aligned}
\frac{\partial \widetilde{q}(z;\widetilde{\rho},\widetilde{m},\widetilde{v})}{\partial \widetilde{\rho}} &= \mathbf{N}(z;\widetilde{m},\widetilde{v})-\delta_0(z) \\
0 = -\frac{d}{d\widetilde{\rho}}\int_{\mathbb{R}} p(z)\log(\widetilde{q}(z))dz &= -\int_{\mathbb{R}} p(z)\frac{\frac{\partial}{\partial\widetilde{\rho}}q(z;\widetilde{\rho},\widetilde{m},\widetilde{v})}{\widetilde{q}(z;\widetilde{\rho},\widetilde{m},\widetilde{v})}dz \\
&= -\int_{\mathbb{R}} p(z)\frac{\mathbf{N}(z;\widetilde{m},\widetilde{v})-\delta_0(z)}{\widetilde{\rho}\mathbf{N}(z;\widetilde{m},\widetilde{v})+(1-\widetilde{\rho})\delta_0(z)}dz \\
&= -\int_{\mathbb{R}} p(z)\frac{1}{\widetilde{\rho}}\frac{\widetilde{\rho}\mathbf{N}(z;\widetilde{m},\widetilde{v})-\widetilde{\rho}\delta_0(z)-\delta_0(z)+\delta_0(z)}{\widetilde{\rho}\mathbf{N}(z;\widetilde{m},\widetilde{v})+(1-\widetilde{\rho})\delta_0(z)}dz \\
&= -\frac{1}{\widetilde{\rho}}\int_{\mathbb{R}} p(z)dz+\frac{1}{\widetilde{\rho}}\int_{\mathbb{R}} p(z)\frac{\delta_0(z)}{\widetilde{\rho}\mathbf{N}(z;\widetilde{m},\widetilde{v})+(1-\widetilde{\rho})\delta_0(z)}dz \\
&\approx -\frac{1}{\widetilde{\rho}}+\frac{1}{\widetilde{\rho}}\lim_{\epsilon\to 0^+}\int_{-\epsilon}^{\epsilon} p(z)\frac{1}{1-\widetilde{\rho}}dz \\
&= -\frac{1}{\widetilde{\rho}}+\frac{1}{\widetilde{\rho}(1-\widetilde{\rho})}\mathbb{P}_p[Z=0],
\end{aligned}$$

yielding the simple and intuitive result:

$$\widetilde{\rho} = 1-\mathbb{P}_p[Z=0].$$

A.1.1 Connection to moment matching

As it turns out, minimizing $KL(p\|q)$ for the spike-and-slab turns out to be equivalent to matching the slab probability and then matching the first and second moments:

$$\begin{aligned}\mathbb{P}_{\widetilde{q}}[Z=0] &= 1-\widetilde{\rho}\\ \mathbb{E}_{\widetilde{q}}[Z] &= \widetilde{\rho}\widetilde{m}\\ \mathbb{V}_{\widetilde{q}}[Z] &= \mathbb{E}_{\widetilde{q}}\left[Z^2\right] - \mathbb{E}_{\widetilde{q}}[Z]^2\\ &= \widetilde{\rho}(\widetilde{m}^2+\widetilde{v}) - \widetilde{\rho}^2\widetilde{m}^2.\end{aligned}$$

Substituting in the values of $(\widetilde{\rho},\widetilde{m},\widetilde{v})$ obtained above in the KL-minimization, we find that we yield the highly intuitive result:

$$\mathbb{P}_p[Z=0]=\mathbb{P}_{\widetilde{q}}[Z=0], \qquad \mathbb{E}_p[Z]=\mathbb{E}_{\widetilde{q}}[Z], \qquad \mathbb{V}_p[Z]=\mathbb{V}_{\widetilde{q}}[Z].$$

A.2 Message parameters following a linear layer

Above, we established the values of parameters $(\widetilde{\rho},\widetilde{m},\widetilde{v})$ in terms of properties of the distribution we intend to approximate. We now shift our focus to deriving the specific values for the transformations that occur within spike-and-slab PBP, namely a linear combination and a ReLU. We begin with the linear combination $z_j^{(\ell,\text{linear})}=\sum_i z_i^{(\ell-1)}w_{ij}$, seeking the values of $\mathbb{E}_p[z_j^{(\ell,\text{linear}}], \mathbb{V}_p[z_j^{(\ell,\text{linear}}]$, and $\mathbb{P}_p[z_j^{(\ell,\text{linear}}=0]$ under the assumption that

$$\begin{aligned}z_i^{(\ell-1)} &\overset{\text{ind}}{\sim} (1-\widetilde{\rho}_i^{(\ell-1)})\delta_0 + \widetilde{\rho}^{(\ell-1)}\text{N}(\widetilde{m}^{(\ell-1)},\widetilde{v}^{(\ell-1)})\\ w_{ij} &\overset{\text{ind}}{\sim} \text{N}(m_{ij},v_{ij}).\end{aligned}$$

From this, we can calculate the mean and variance

$$\begin{aligned}\mathbb{E}\left[z_j^{(\ell,\text{linear})}\right] &= \sum_i(\widetilde{\rho}_i^{(\ell-1)}\widetilde{m}_i^{(\ell)})m_{ij}\\ \mathbb{V}\left[z^{(\ell,\text{linear})}\right] &= \sum_i \widetilde{\rho}_i^{(\ell-1)}(\widetilde{m}_i^{(\ell-1)})^2v_{ij} + \widetilde{\rho}_i^{(\ell-1)}\widetilde{v}_i^{(\ell-1)}m_{ij}^2\\ &\quad + \widetilde{\rho}_i^{(\ell-1)}\widetilde{v}_i^{(\ell-1)}v_{ij} + \widetilde{\rho}_i^{(\ell-1)}(1-\widetilde{\rho}_i^{(\ell-1)})(\widetilde{m}_i^{(\ell-1)})^2m_{ij}^2.\end{aligned}$$

Now, incorporating the rescaling transformation used by Hernández-Lobato and Adams (2015), $\mathbf{z}^{(\ell,\text{linear})}=\mathbf{W}^{(\ell)}\mathbf{z}^{(\ell-1)}/\sqrt{n_{\ell-1}+1}$ we have

$$\begin{aligned}\mathbb{E}\left[\mathbf{z}^{(\ell,\text{linear})}\right] &= \mathbf{M}^{(\ell)}\left(\widetilde{\rho}^{(\ell-1)}\circ\widetilde{\mathbf{m}}^{(\ell-1)}\right)/\sqrt{n_{\ell-1}+1}\\ \mathbb{V}\left[\mathbf{z}^{(\ell,\text{linear})}\right] &= \Big[\mathbf{V}^{(\ell)}\left(\widetilde{\rho}^{(\ell-1)}\circ\widetilde{\mathbf{m}}^{(\ell-1)}\circ\widetilde{\mathbf{m}}^{(\ell-1)}\right) + \left(\mathbf{M}^{(\ell)}\circ\mathbf{M}^{(\ell)}\right)\left(\widetilde{\rho}^{(\ell-1)}\circ\widetilde{\mathbf{v}}^{(\ell-1)}\right)\\ &\quad + \left(\mathbf{M}^{(\ell)}\circ\mathbf{M}^{(\ell)}\right)\left(\widetilde{\rho}^{(\ell-1)}\circ\left(1-\widetilde{\rho}^{(\ell-1)}\right)\circ\widetilde{\mathbf{m}}^{(\ell-1)}\circ\widetilde{\mathbf{m}}^{(\ell-1)}\right)\\ &\quad + \mathbf{V}^{(\ell)}\left(\widetilde{\rho}^{(\ell-1)}\circ\widetilde{\mathbf{v}}^{(\ell-1)}\right)\Big]\Big/(n_{\ell-1}+1)\\ \mathbb{P}\left[\mathbf{z}^{(\ell,\text{linear})}=0\right] &= \prod_i(1-\widetilde{\rho}_i^{(\ell-1)}).\end{aligned}$$

Combining these moments with the general forms derived earlier, we have the following equations (replacing Equations 13-14 of the original PBP method) for the distribution of linear layers in SSPBP with parameters $\widetilde{\boldsymbol{\rho}}^{(\ell,\text{linear})}, \widetilde{\mathbf{m}}^{(\ell,\text{linear})}, \widetilde{\mathbf{v}}^{(\ell,\text{linear})}$:

$$
\begin{aligned}
\widetilde{\boldsymbol{\rho}}_j^{(\ell,\text{linear})} &= 1 - \prod_i (1 - \widetilde{\boldsymbol{\rho}}_i^{(\ell-1)}) \\
\widetilde{\boldsymbol{\rho}}^{(\ell,\text{linear})} \circ \widetilde{\mathbf{m}}^{(\ell,\text{linear})} &= \frac{\mathbf{M}^{(\ell}(\widetilde{\boldsymbol{\rho}}^{(\ell-1)} \circ \widetilde{\mathbf{m}}^{(\ell-1)})}{\sqrt{n_{\ell-1}+1}} \\
\widetilde{\boldsymbol{\rho}}^{(\ell,\text{linear})} \circ \widetilde{\boldsymbol{\rho}}^{(\ell,\text{linear})} \circ \widetilde{\mathbf{v}}^{(\ell,\text{linear})} &= \widetilde{\boldsymbol{\rho}}^{(\ell,\text{linear})} \circ \mathbb{V}\left[z^{(\ell,\text{linear})}\right] - (1 - \widetilde{\boldsymbol{\rho}}^{(\ell,\text{linear})}) \mathbb{E}\left[\mathbf{z}^{(\ell,\text{linear})}\right]^2 \\
\widetilde{\boldsymbol{\rho}}^{(\ell,\text{linear})} \circ (n_{\ell-1}+1) \widetilde{\mathbf{v}}^{(\ell,\text{linear})} &= \\
&\quad \mathbf{V}^{(\ell)} \left(\widetilde{\boldsymbol{\rho}}^{(\ell-1)} \circ \widetilde{\mathbf{m}}^{(\ell-1)} \circ \widetilde{\mathbf{m}}^{(\ell-1)}\right) \\
&\quad + \left(\mathbf{V}^{(\ell)} + \mathbf{M}^{(\ell)} \circ \mathbf{M}^{(\ell)}\right) \left(\widetilde{\boldsymbol{\rho}}^{(\ell-1)} \circ \widetilde{\mathbf{v}}^{(\ell-1)}\right) \\
&\quad + \left(\mathbf{M}^{(\ell)} \circ \mathbf{M}^{(\ell)}\right) \left(\widetilde{\boldsymbol{\rho}}^{(\ell-1)} \circ \left(1 - \widetilde{\boldsymbol{\rho}}^{(\ell-1)}\right) \circ \widetilde{\mathbf{m}}^{(\ell-1)} \circ \widetilde{\mathbf{m}}^{(\ell-1)}\right).
\end{aligned}
\tag{6}
$$

A.3 Message parameters following a linear layer plus ReLU activation

We now consider passing the messages from Equation 6 through a ReLU, to get the parameters $(\widetilde{\boldsymbol{\rho}}^{(\ell,\text{ReLU})}, \widetilde{\mathbf{m}}^{(\ell,\text{ReLU})}, \widetilde{\mathbf{v}}^{(\ell,\text{ReLU})})$ of the resulting message.

To compute these parameters, it's useful to employ a hierarchical model:

$$
\begin{aligned}
A_i &\sim (1 - \widetilde{\rho}_i^{(\ell,\text{linear})})\delta_0 + \widetilde{\rho}_i^{(\ell,\text{linear})} \text{N}(\widetilde{m}^{(\ell,\text{linear})}, \widetilde{v}^{(\ell,\text{linear})}) \\
T_i &\sim \text{Ber}(\widetilde{\rho}_i^{(\ell,\text{linear})}) \\
A_i | T_i = 0 &\sim \delta_0 \\
A_i | T_i = 1 &\sim \text{N}(\widetilde{m}_i^{(\ell,\text{linear})}, \widetilde{v}_i^{(\ell,\text{linear})}) \\
B_i | A_i = a &= \text{ReLU}(a) = \max(a, 0),
\end{aligned}
$$

implying

$$
\begin{aligned}
B_i \sim &\left(1 - \widetilde{\rho}_i^{(\ell,\text{linear})} \Phi\left(\frac{\widetilde{m}_i^{(\ell,\text{linear})}}{\sqrt{\widetilde{v}_i^{(\ell,\text{linear})}}}\right)\right)\delta_0 \\
&+ \widetilde{\rho}_i^{(\ell,\text{linear})} \Phi\left(\frac{\widetilde{m}_i^{(\ell,\text{linear})}}{\sqrt{\widetilde{v}_i^{(\ell,\text{linear})}}}\right) \text{TN}_{(0,\infty)}\left(\widetilde{m}_i^{(\ell,\text{linear})}, \widetilde{v}_i^{(\ell,\text{linear})}\right)
\end{aligned}
$$

where TN indicates a truncated normal. Similarly, let's now introduce a hierarchy for B_i, along with a "dummy" variable X_i to make the notation below a little more straightforward:

$$
\begin{aligned}
L_i &\sim \text{Ber}\left(\widetilde{\rho}_i^{(\ell,\text{linear})} \Phi\left(\frac{\widetilde{m}_i^{(\ell,\text{linear})}}{\sqrt{\widetilde{v}_i^{(\ell,\text{linear})}}}\right)\right) \\
B_i | L_i = 0 &\sim \delta_0 \\
X_i &\sim \text{TN}_{(0,\infty)}\left(\widetilde{m}_i^{(\ell,\text{linear})}, \widetilde{v}_i^{(\ell,\text{linear})}\right) \\
B_i | L_i = 1 &\sim \delta_{X_i}.
\end{aligned}
$$

Thus, $\mathbb{E}[X_i], \mathbb{V}[X_i]$, *etc*. correspond to the conditional expectation and variance $\mathbb{E}[B_i | L_i = 1], \mathbb{V}[B_i | L_i = 1]$.

We next compute the moments of $(\widetilde{\boldsymbol{\rho}}^{(\ell,\text{ReLU})}, \widetilde{\mathbf{m}}^{(\ell,\text{ReLU})}, \widetilde{\mathbf{v}}^{(\ell,\text{ReLU})})$ by the moment-matching we derived above:

$$\begin{aligned}
\mathbb{P}[B_i = 0] =& \mathbb{P}[L_i = 0] = 1 - \widetilde{\rho}_i^{(\ell,\text{linear})} \Phi\left(\widetilde{m}_i^{(\ell,\text{linear})} \Big/ \sqrt{\widetilde{v}_i^{(\ell,\text{linear})}}\right) \\
\mathbb{E}[B_i] =& \mathbb{E}\left[\mathbb{E}[B_i | L_i]\right] = \mathbb{P}[L_i = 0] \cdot 0 + \mathbb{P}[L_i = 1]\mathbb{E}[X_i] \\
=& \mathbb{P}[L_i = 1]\mathbb{E}[X_i] = \widetilde{\rho}_i^{(\ell,\text{linear})} \Phi\left(\widetilde{m}_i^{(\ell,\text{linear})} \Big/ \sqrt{\widetilde{v}_i^{(\ell,\text{linear})}}\right) \mathbb{E}[X_i] \\
\mathbb{V}[B_i] =& \mathbb{E}[B_i^2] - \mathbb{E}[B_i]^2 = \mathbb{P}[L_i = 0] \cdot 0 + \mathbb{P}[L_i = 1]\mathbb{E}[X_i^2] - \left(\mathbb{P}[L_i = 1]\mathbb{E}[X_i]\right)^2 \\
=& \mathbb{P}[L_i = 1]\mathbb{E}[X_i^2] - \mathbb{P}[L_i = 1]^2 \mathbb{E}[X_i]^2 \\
=& \widetilde{\rho}_i^{(\ell,\text{linear})} \Phi\left(\widetilde{m}_i^{(\ell,\text{linear})} \Big/ \sqrt{\widetilde{v}_i^{(\ell,\text{linear})}}\right) \mathbb{E}[X^2] \\
& - \left(\widetilde{\rho}_i^{(\ell,\text{linear})}\right)^2 \Phi\left(\widetilde{m}_i^{(\ell,\text{linear})} \Big/ \sqrt{\widetilde{v}_i^{(\ell,\text{linear})}}\right)^2 \mathbb{E}[X_i]^2.
\end{aligned}$$

Using the spike-and-slab moment-matching results above, we obtain:

$$\begin{aligned}
\widetilde{\rho}_i^{(\ell,\text{ReLU})} &= 1 - \mathbb{P}[B_i = 0] = 1 - \left(1 - \widetilde{\rho}_i^{(\ell,\text{linear})} \Phi\left(\widetilde{m}_i^{(\ell,\text{linear})} \Big/ \sqrt{\widetilde{v}_i^{(\ell,\text{linear})}}\right)\right) = \rho \Phi\left(\frac{m}{\sqrt{v}}\right) \\
\widetilde{m}_i^{(\ell,\text{ReLU})} &= \frac{\mathbb{E}[B_i]}{\widetilde{\rho}_i^{(\ell,\text{ReLU})}} = \frac{\widetilde{\rho}_i^{(\ell,\text{linear})} \Phi\left(\widetilde{m}_i^{(\ell,\text{linear})} \Big/ \sqrt{\widetilde{v}_i^{(\ell,\text{linear})}}\right) \mathbb{E}[X_i]}{\widetilde{\rho}_i^{(\ell,\text{linear})} \Phi\left(\widetilde{m}_i^{(\ell,\text{linear})} \Big/ \sqrt{\widetilde{v}_i^{(\ell,\text{linear})}}\right)} = \mathbb{E}[X_i] \\
\widetilde{v}_i^{(\ell,\text{ReLU})} &= \frac{\mathbb{V}[B_i] - \widetilde{\rho}_i^{(\ell,\text{ReLU})}\left(1 - \widetilde{\rho}_i^{(\ell,\text{ReLU})}\right)\left(\widetilde{m}_i^{(\ell,\text{ReLU})}\right)^2}{\widetilde{\rho}_i^{(\ell,\text{ReLU})}} = \mathbb{E}[X_i^2] - \mathbb{E}[X_i]^2 = \mathbb{V}[X_i].
\end{aligned}$$

In other words, we match the probability mass of the spike and match the mean and variance in our normal approximation to the mean and variance of the truncated normal! This yields our final message parameters,

$$\begin{aligned}
\widetilde{m}_i^{(\ell,\text{ReLU})} &= \widetilde{m}_i^{(\ell,\text{linear})} + \gamma_i \sqrt{\widetilde{v}_i^{(\ell,\text{linear})}} \\
\widetilde{v}_i^{(\ell,\text{ReLU})} &= \widetilde{v}_i^{(\ell,\text{linear})} \left(1 - \gamma_i \left(\gamma_i + \frac{\widetilde{m}_i^{(\ell,\text{linear})}}{\sqrt{\widetilde{v}_i^{(\ell,\text{linear})}}}\right)\right) \\
\widetilde{\rho}_i^{(\ell,\text{ReLU})} &= \widetilde{\rho}_i^{(\ell,\text{linear})} \Phi\left(\frac{\widetilde{m}_i^{(\ell,\text{linear})}}{\sqrt{\widetilde{v}_i^{(\ell,\text{linear})}}}\right) \\
\gamma_i &= \frac{\phi\left(\frac{\widetilde{m}_i^{(\ell,\text{linear})}}{\sqrt{\widetilde{v}_i^{(\ell,\text{linear})}}}\right)}{\Phi\left(\frac{\widetilde{m}_i^{(\ell,\text{linear})}}{\sqrt{\widetilde{v}_i^{(\ell,\text{linear})}}}\right)}.
\end{aligned}$$

The intermediate parameter γ_i can be replaced with a "robust" version as appropriate, following (Hernández-Lobato and Adams, 2015).

A.4 Normalization Constant

The normalization constant Z of Hernández-Lobato and Adams (2015) in Equation 12 (and its uses elsewhere in the automatic differentiation for the model parameters) is modified slightly in our

approach. As final output parameters, we produce $(\widetilde{\rho}^{(L)}, \widetilde{m}^{(L)}, \widetilde{v}^{(L)})$, along with the homoscedastic noise estimate $\beta^\gamma/(\alpha^\gamma - 1)$ (unchanged from PBP). As such, the relevant replacement for Equation 12 of (Hernández-Lobato and Adams, 2015) is

$$Z \approx (1 - \widetilde{\rho}^{(L)})\mathrm{N}(y_n|0, \beta^\gamma/(\alpha^\gamma - 1)) + \rho^{(L)}\mathrm{N}(y_n|\widetilde{m}^{(L)}, \beta^\gamma/(\alpha^\gamma - 1) + \widetilde{v}^{(L)}).$$

B In the presence of a bias term, PBP is numerically equivalent to SSPBP

Here, we show that after the linear combination step, the models produce the same resultant distribution. Consider the following model for the output Y of a single node (for simplicity in notation) with random K-dimensional input $\mathbf{x}$, noting that this corresponds exactly to the first hidden layer's activation function and the second layer's linear combination step[5]:

$$\begin{aligned}
x_i &\sim \mathrm{N}(\widetilde{m}_i, \widetilde{v}_i), \quad i = 1:K \\
t_i|x_i &= \mathrm{ReLU}(x_i), \quad i = 1:K \\
t_{K+1} &\sim \delta_1 = \mathrm{N}(1, 0) \quad \leftarrow \text{the bias term} \\
w_i &\sim \mathrm{N}(m_i, v_i), \quad i = 1:K+1 \\
y|\mathbf{t}, \mathbf{w} &= \frac{1}{\sqrt{K+1}}\mathbf{w}^\top \mathbf{t}.
\end{aligned}$$

Following Hernández-Lobato and Adams (2015), we also assume

$$\begin{aligned}
\alpha_i &= \frac{\widetilde{m}_i}{\sqrt{\widetilde{v}_i}} \\
\gamma_i &= \frac{\phi(\alpha_i)}{\Phi(\alpha_i)} \\
v_i' &= \widetilde{m}_i + \widetilde{v}_i\gamma_i.
\end{aligned}$$

B.1 Parameters of the PBP message

Under standard PBP, we model the distribution of t_i with a Gaussian with mean $m_i^{(t)}$ and variance $v_i^{(t)}$, where

$$\begin{aligned}
m_i^{(t)} &= \Phi(\alpha_i)v_i', \quad i = 1:K, \quad m_{K+1}^{(t)} = 1, \\
v_i^{(t)} &= (1 - \Phi(\alpha_i))m_i^{(t)}v_i' + \Phi(\alpha_i)v_i(1 - \gamma_i(\gamma_i + \alpha_i)), \quad i = 1:K, \\
&= \Phi(\alpha_i)(1 - \Phi(\alpha_i))(v_i')^2 + \Phi(\alpha_i)v_i(1 - \gamma_i(\gamma_i + \alpha_i)), \quad v_{K+1}^{(t)} = 0.
\end{aligned}$$

We then approximate the distribution of y using a Gaussian with mean $m^{(y)}$ and variance $v^{(y)}$, where

$$\begin{aligned}
m^{(y)} &= \frac{1}{\sqrt{K+1}}\mathbf{m}^\top\mathbf{m}^{(t)} = \frac{1}{\sqrt{K+1}}\left(m_{K+1} + \sum_{i=1}^K m_i\Phi(\alpha_i)v_i'\right) \\
(K+1)v^{(y)} &= (\mathbf{m} \circ \mathbf{m} + \mathbf{v})^\top \mathbf{v}^{(t)} + \mathbf{v}^\top(\mathbf{m}^{(t)} \circ \mathbf{m}^{(t)}) \\
&= v_{K+1} + \sum_{i=1}^K (m_i^2 + v_i)(1 - \Phi(\alpha_i))\Phi(\alpha_i)(v_i')^2 \\
&\quad + (m_i^2 + v_i)\Phi(\alpha_i)\widetilde{v}_i(1 - \gamma_i(\gamma_i + \alpha_i)) + v_i\Phi(\alpha_i)^2(v_i')^2.
\end{aligned} \tag{7}$$

[5]The factor $1/\sqrt{K+1}$ "keeps the scale of the input to each neuron independent of the number of incoming connections." (Hernández-Lobato and Adams, 2015)

Table 5: Mean and standard error of average test set RMSE of PBP and SSPBP, on eight datasets.

	1 Layer 10 Nodes		1 Layer 100 Nodes	
Dataset	**PBP**	**SSPBP**	**PBP**	**SSPBP**
Boston Housing	**3.787±0.344**	3.789±0.351	**3.432±0.244**	3.437±0.255
Combined Cycle Power Plant	**4.057±0.046**	4.068±0.042	**4.157±0.025**	4.159±0.024
Concrete Compression Strength	**6.221±0.222**	6.426±0.184	**5.565±0.056**	5.574±0.028
Energy Efficiency	2.100±0.082	**2.086±0.077**	**1.879±0.063**	1.898±0.069
Kin8nm	0.140±0.002	**0.137±0.003**	**0.091±0.000**	**0.091±0.001**
Naval Propulsion	**0.009±0.000**	**0.009±0.000**	**0.004±0.000**	**0.004±0.000**
Wine Quality Red	0.619±0.010	**0.617±0.009**	0.630±0.014	**0.628±0.014**
Yacht Hydrodynamics	1.769±0.197	**1.747±0.177**	1.220±0.057	**1.176±0.054**

B.2 Parameters of the SSPBP message

We repeat our analysis using spike-and-slab approximations, so the distribution of t_i is parameterized by $(\rho_i^{(t)}, m_i^{(t)}, v_i^{(t)})$, and the distribution of y is parameterized by $(\rho^{(y)}, m^{(y)}, v^{(y)})$, such that

$$\begin{aligned} \rho_i^{(t)} &= \Phi(\alpha_i), \quad i = 1:K, & \rho_{K+1}^{(t)} &= 1, \\ m_i^{(t)} &= v_i', \quad i = 1:K, & m_{K+1}^{(t)} &= 1, \\ v_i^{(t)} &= \widetilde{v}_i(1 - \gamma_i(\gamma_i + \alpha_i)), \ \ i = 1:K & v_{K+1}^{(t)} &= 0, \end{aligned}$$

and

$$\begin{aligned} \rho^{(y)} &= 1 - \prod_{i=1}^{K+1}(1 - \rho_i^{(t)}) = 1 - \prod_{i=1}^{K}(1 - \Phi(\alpha_i))(1-1) = 1 \\ m^{(y)} &= \frac{1}{\sqrt{K+1}} \sum_{i=1}^{K+1} m_i \rho_i^{(t)} m_i^{(t)} \\ &= \frac{1}{\sqrt{K+1}} \left(m_{K+1} + \sum_{i=1}^{K} m_i \Phi(\alpha_i) v_i' \right) \\ (K+1)v^{(y)} &= -\rho^{(y)}(1-\rho^{(y)})(m^{(y)})^2 + \mathbf{v}^\top(\boldsymbol{\rho}^{(t)} \circ \mathbf{m}^{(t)} \circ \mathbf{m}^{(t)}) \\ &\quad + (\mathbf{m} \circ \mathbf{m} + \mathbf{v})^\top(\boldsymbol{\rho}^{(t)} \circ \boldsymbol{v}^{(t)}) \\ &\quad + (\mathbf{m} \circ \mathbf{m})^\top(\boldsymbol{\rho}^{(t)} \circ (1 - \boldsymbol{\rho}^{(t)}) \circ \mathbf{m}^{(t)} \circ \mathbf{m}^{(t)}) \\ &= v_{K+1} + \sum_{i=1}^{K}(m_i^2 + v_i)(1 - \Phi(\alpha_i))\Phi(\alpha_i)(v_i')^2 \\ &\quad + (m_i^2 + v_i)\Phi(\alpha_i)\widetilde{v}_i(1 - \gamma_i(\gamma_i + \alpha_i)) + v_i\Phi(\alpha_i)^2(v_i')^2. \end{aligned} \tag{8}$$

Comparing Equations 7 and 8, we see the two methods are exactly equivalent after a linear combination step, with slab probability equal to 1 in the spike-and-slab variant. As such, any future hidden layers will behave identically. In the regression setting of PBP, the final transformation is solely a linear combination, so the final output is the same.

C Additional Empirical Results

We include here additional results for RMSE and log-likelihood of different model configurations. Table 5 and Table 6 report the RMSE and log-likelihood for the standard version of PBP and SSPBP, respectively. Similarly, Table 7 and Table 8 report the RMSE and log-likelihood of the "bias-free" versions of PBP and SSPBP, respectively.

Table 6: Mean and standard error of the test set log-likelihood of PBP and SSPBP, on eight datasets.

	1 Layer 10 Nodes		**1 Layer 100 Nodes**	
Dataset	**PBP**	**SSPBP**	**PBP**	**SSPBP**
Boston Housing	**-2.810±0.140**	-2.823±0.141	**-2.727±0.106**	-2.725±0.111
Combined Cycle Power Plant	**-2.821±0.010**	-2.824±0.010	-2.844±0.006	**-2.845±0.006**
Concrete Compression Strength	**-3.253±0.039**	-3.284±0.032	-3.136±0.011	**-3.138±0.006**
Energy Efficiency	-2.179±0.053	**-2.169±0.047**	-2.056±0.034	**-2.066±0.038**
Kin8nm	0.549±0.017	**0.567±0.021**	**0.968±0.003**	0.975±0.005
Naval Propulsion	3.275±0.008	**3.276±0.005**	3.949±0.005	**3.945±0.006**
Wine Quality Red	-0.941±0.017	**-0.937±0.015**	**-0.959±0.027**	-0.955±0.026
Yacht Hydrodynamics	-2.022±0.083	**-2.012±0.070**	**-1.751±0.017**	-1.739±0.011
	1 Layer 50 Nodes		**2 Layers 10 Nodes**	
Dataset	**PBP**	**SSPBP**	**PBP**	**SSPBP**
Boston Housing	**-2.771±0.076**	-2.787±0.087	**-2.576±0.073**	-2.535±0.076
Combined Cycle Power Plant	**-2.834±0.009**	**-2.834±0.008**	-2.828±0.017	**-2.830±0.016**
Concrete Compression Strength	**-3.149±0.025**	-3.157±0.022	**-3.218±0.028**	-3.201±0.029
Energy Efficiency	-2.049±0.042	**-2.047±0.041**	-1.802±0.025	**-1.878±0.047**
Kin8nm	**0.901±0.008**	0.885±0.010	**0.784±0.034**	0.799±0.015
Naval Propulsion	**3.725±0.006**	3.717±0.007	3.713±0.019	**3.686±0.005**
Wine Quality Red	-1.002±0.006	**-1.001±0.006**	**-1.009±0.025**	-1.003±0.016
Yacht Hydrodynamics	**-1.767±0.027**	-1.775±0.031	-1.714±0.054	**-1.749±0.032**

Table 7: Mean and standard error of average test set RMSE of the bias-free versions of PBP and SSPBP, on eight datasets.

	1 Layer 10 Nodes		**1 Layer 100 Nodes**	
Dataset	**PBP**	**SSPBP**	**PBP**	**SSPBP**
Boston Housing	4.120±0.104	**4.061±0.149**	3.279±0.146	**3.260±0.124**
Combined Cycle Power Plant	**4.330±0.033**	4.335±0.026	4.208±0.049	**4.207±0.047**
Concrete Compression Strength	**8.240±0.081**	8.242±0.102	6.726±0.190	**6.718±0.183**
Energy Efficiency	2.060±0.059	**2.054±0.067**	**1.901±0.051**	**1.901±0.052**
Kin8nm	**0.165±0.001**	**0.165±0.001**	**0.156±0.002**	**0.156±0.002**
Naval Propulsion	**0.009±0.000**	**0.009±0.000**	**0.005±0.000**	**0.005±0.000**
Wine Quality Red	0.642±0.003	**0.636±0.004**	0.637±0.008	**0.635±0.007**
Yacht Hydrodynamics	**6.248±0.381**	6.275±0.353	7.233±0.486	**6.215±0.152**

Table 8: Mean and standard error of the test set log-likelihood of bias-free versions of PBP and SSPBP, on eight datasets.
† Due to numerical issues, the five trials for this model were repeated with a different random seed.
‡ Due to numerical issues, we report results from seven trails out of ten that yielded finite values for the test set log-likelihood.

	1 Layer 10 Nodes		**1 Layer 100 Nodes**	
Dataset	**PBP**	**SSPBP**	**PBP**	**SSPBP**
Boston Housing	-2.912±0.051	**-2.891±0.057**	**-2.597±0.048**	-2.589±0.040
Combined Cycle Power Plant	**-2.885±0.008**	-2.886±0.006	**-2.857±0.012**	-2.856±0.011
Concrete Compression Strength	**-3.551±0.015**	-3.555±0.018	**-3.327±0.029**	-3.326±0.028
Energy Efficiency	-2.151±0.033	**-2.148±0.038**	-2.067±0.026	**-2.068±0.027**
Kin8nm	0.382±0.005	**0.395±0.003**†	**0.441±0.014**	**0.441±0.013**
Naval Propulsion	**3.251±0.008**	3.239±0.008	**3.885±0.006**	3.895±0.012
Wine Quality Red	-0.978±0.006	**-0.969±0.007**	**-0.969±0.013**	-0.965±0.013
Yacht Hydrodynamics	**-3.285±0.079**	-3.291±0.076	**-3.335±0.046**	-3.213±0.027

	1 Layer 50 Nodes		**2 Layers 10 Nodes**	
Dataset	**PBP**	**SSPBP**	**PBP**	**SSPBP**
Boston Housing	**-2.699±0.105**	-2.709±0.108	**-2.944±0.191**	-2.916±0.146
Combined Cycle Power Plant	-2.879±0.013	**-2.878±0.012**	**-2.852±0.004**	-2.851±0.006
Concrete Compression Strength	-3.385±0.019	**-3.373±0.026**	**-3.343±0.034**	-3.349±0.033‡
Energy Efficiency	-2.011±0.017	**-2.010±0.019**	**-1.902±0.018**	-1.850±0.021
Kin8nm	0.421±0.013	**0.424±0.012**	**0.655±0.009**	0.662±0.005†
Naval Propulsion	**3.673±0.006**	3.658±0.006	3.772±0.029	**3.703±0.021**
Wine Quality Red	-0.944±0.023	**-0.939±0.022**	-0.974±0.022	**-0.986±0.025**
Yacht Hydrodynamics	**-3.264±0.055**	-3.290±0.059	-2.687±0.054	**-2.730±0.068**