Uncertainty propagation in neural networks for sparse coding

Danil Kuzin[†], **Olga Isupova**[⋆], **Lyudmila Mihaylova**[†]
[†]Department of Automatic Control and System Engineering, University of Sheffield, UK
[⋆]Department of Engineering Science, University of Oxford, UK
{dkuzin1,l.s.mihaylova}@sheffield.ac.uk,olga.isupova@eng.ox.ac.uk

1 Introduction

The idea of Bayesian learning in neural networks (NNs) [1] has recently gained an attention with the development of distributed approximate inference techniques [2, 3] and general boost in popularity of deep learning. Recently several techniques [4, 5] have been proposed to handle specific types of NNs with efficient Bayesian inference. For example, feed-forward networks with the rectified linear unit nonlinearity [6], networks with discrete distributions [7], recurrent networks [8].

In this paper, we consider the area of sparse coding. The sparse coding problem can be viewed as a linear regression problem with the additional assumption that the majority of the basis representation coefficients should be zeros. This sparsity assumption may be represented as $l1$ penalty [9], or, in Bayesian interpretation, as a prior that has a sharp peak at zero [10]. One of the modern approaches for sparse coding utilises NNs with the soft-thresholding nonlinearity [11, 12]. Sparse coding is widely used in different applications, such as compressive sensing [13], image and video processing [14, 15], neuroscience [16, 17].

A novel method to propagate uncertainty through the soft-thresholding nonlinearity is proposed in this paper. At every layer the current distribution of the target vector is represented as a spike and slab distribution [18], which represents the probabilities of each variable being zero, or Gaussian-distributed. Using the proposed method of uncertainty propagation, the gradients of the logarithms of normalisation constants are derived, that can be used to update a weight distribution. A novel Bayesian NN for sparse coding is designed utilising both the proposed method of uncertainty propagation and Bayesian inference algorithm.

The main contributions of this paper are: (*i*) for the first time a method for uncertainty propagation through the soft-thresholding nonlinearity is proposed for a Bayesian NN; (*ii*) an efficient posterior inference algorithm for weights and outputs of NNs with the soft-thresholding nonlinearity is developed; (*iii*) a novel Bayesian NN for sparse coding is designed.

The rest of the paper is organised as follows. A NN approach for sparse coding is described in Section 2.1. The Bayesian formulation is introduced in Section 2.2. Section 3 provides the experimental results. The proposed forward uncertainty propagation and probabilistic backpropagation methods are given in Appendices A and B.

2 Neural networks for sparse coding

This section presents background knowledge about networks for sparse coding and then describes the novel Bayesian neural network.

Third workshop on Bayesian Deep Learning (NeurIPS 2018), Montréal, Canada.

2.1 Frequentist neural networks

The NN approach to sparse coding is based on earlier Iterative Shrinkage and Thresholding Algorithm (ISTA) [19]. It addresses the sparse coding problem as the linear regression problem with the $l1$ penalty that promotes sparsity. For the linear regression model with observations $\mathbf{y} \in \mathbb{R}^K$, the design matrix $\mathbf{X} \in \mathbb{R}^{K \times D}$, and the sparse unknown vector of weights $\boldsymbol{\beta} \in \mathbb{R}^D$, ISTA minimises

$$||\mathbf{X}\boldsymbol{\beta} - \mathbf{y}||_2^2 + \alpha||\boldsymbol{\beta}||_1 \text{ w.r.t. } \boldsymbol{\beta}, \tag{1}$$

where α is a regularisation parameter.

At every iteration l, ISTA obtains the new estimate $\widehat{\boldsymbol{\beta}}_l$ of the target vector $\boldsymbol{\beta}$ as the linear transformation $\mathbf{b} = \mathbf{W}\mathbf{y} + \mathbf{S}\widehat{\boldsymbol{\beta}}_{l-1}$ propagated through the soft-thresholding function

$$h_\lambda(\mathbf{b}) = \text{sgn}(\mathbf{b}) \max(|\mathbf{b}| - \lambda, 0), \tag{2}$$

where λ is a shrinkage parameter. In ISTA, weights $\mathbf{W}$ and $\mathbf{S}$ of the linear transformation are assumed fixed.

Algorithm 1 LISTA forward propagation

Input: observations $\mathbf{y}$, weights $\mathbf{W}, \mathbf{S}$, number of layers L

Dense layer $\mathbf{b} \leftarrow \mathbf{W}\mathbf{y}$
Soft-thresholding function $\widehat{\boldsymbol{\beta}}_0 \leftarrow h_\lambda(\mathbf{b})$
for $l = 1$ **to** L **do**
    Dense layer $\mathbf{c}_l \leftarrow \mathbf{b} + \mathbf{S}\widehat{\boldsymbol{\beta}}_{l-1}$
    Soft-thresholding function $\widehat{\boldsymbol{\beta}}_l \leftarrow h_\lambda(\mathbf{c}_l)$
end for
Output: $\widehat{\boldsymbol{\beta}} \leftarrow \widehat{\boldsymbol{\beta}}_L$

In contrast to ISTA, Learned ISTA (LISTA) [11] learns the values of matrices $\mathbf{W}$ and $\mathbf{S}$ based on a set of pairs $\{\mathbf{Y}, \mathbf{B}\} = \{\mathbf{y}^{(n)}, \boldsymbol{\beta}^{(n)}\}_{n=1}^N$, where N is the number of these pairs. To achieve this, ISTA is limited with the fixed amount of iterations L and interpreted as a recurrent NN: every iteration l of ISTA corresponds to the layer l of LISTA. A vector $\widehat{\boldsymbol{\beta}}$ for an observation $\mathbf{y}$ is predicted by Algorithm 1.

2.2 BayesLISTA

This section introduces the proposed Bayesian version of LISTA (BayesLISTA). The prior distributions are imposed on the unknown weights

$$p(\mathbf{W}) = \prod_{d=1}^{D}\prod_{k=1}^{K} \mathcal{N}(w_{ij}; 0, \eta^{-1}), \quad p(\mathbf{S}) = \prod_{d'=1}^{D}\prod_{d''=1}^{D} \mathcal{N}(s_{d'd''}; 0, \eta^{-1}), \tag{3}$$

where η is the precision of the Gaussian distribution.

For every layer l of BayesLISTA, $\widehat{\boldsymbol{\beta}}_l$ is assumed to have the spike and slab distribution with the spike probability $\boldsymbol{\omega}$, the slab mean $\mathbf{m}$, and the slab variance $\mathbf{v}$

$$[\widehat{\boldsymbol{\beta}}_l]_d \sim \omega_d\delta_0 + (1 - \omega_d)\mathcal{N}(m_d, v_d), \tag{4}$$

where δ_0 is the delta-function that represents a spike, $[\cdot]_d$ denotes the d-th component of a vector. In appendix we show that the output of the next layer $\widehat{\boldsymbol{\beta}}_{l+1}$ can be approximated with the spike and slab distribution and, therefore, the output of the BayesLISTA network $\widehat{\boldsymbol{\beta}}$ has the spike and slab distribution.

To introduce the uncertainty of predictions, we assume that the true $\boldsymbol{\beta}$ is an output $f(\mathbf{y}; \mathbf{S}, \mathbf{W}, \lambda)$ of the BayesLISTA network corrupted by the additive Gaussian zero-mean noise with the precision γ. Then the likelihood of $\mathbf{B}$ is defined as

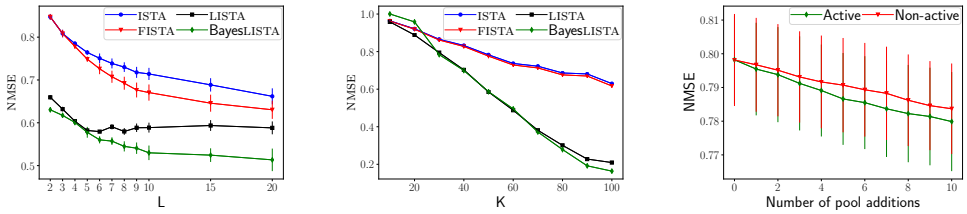
$$p(\mathbf{B}|\mathbf{Y}, \mathbf{W}, \mathbf{S}, \gamma, \lambda) = \prod_{n=1}^{N}\prod_{d=1}^{D} \mathcal{N}\left(\beta_d^{(n)}; [f(\mathbf{y}; \mathbf{S}, \mathbf{W}, \lambda)]_d, \gamma^{-1}\right) \tag{5}$$

Gamma prior distributions with parameters $a^\cdot$ and $b^\cdot$ are specified on the introduced Gaussian precisions

$$p(\gamma) = \text{Gam}\left(\gamma; a^\gamma, b^\gamma\right), \qquad p(\eta) = \text{Gam}\left(\eta; a^\eta, b^\eta\right) \tag{6}$$

The posterior distribution is then

$$p(\mathbf{W}, \mathbf{S}, \gamma, \eta|\mathbf{B}, \mathbf{Y}, \lambda) = \frac{p(\mathbf{B}|\mathbf{Y}, \mathbf{W}, \mathbf{S}, \gamma, \lambda)p(\mathbf{W}|\eta)p(\mathbf{S}|\eta)p(\eta)p(\gamma)}{p(\mathbf{B}|\mathbf{Y}, \lambda)} \tag{7}$$



(a) Synthetic for different L (b) Synthetic for different K (c) Active learning example

Figure 1: NMSE results. The synthetic data results for different number of layers (a) and for different sizes of observations (b). The active learning example results on the MNIST data (c).

The shrinkage parameter λ is a hyperparameter of the model.

In the appendix we describe modification of LISTA forward propagation (Algorithm 1) to include probability distributions of the random variables introduced in this section and also an efficient Bayesian inference algorithm.

3 Experiments

Proposed BayesLISTA is evaluated on sparse coding problems and compared with LISTA [11], ISTA [19] and Fast ISTA (FISTA) [20]. The number of iterations in ISTA and FISTA and the number of layers in NNs is L. For quantitative comparison the normalised mean square error (NMSE) is used.

3.1 Predictive performance on synthetic data

First, performance is analysed on synthetic data. We generate $N_{\text{train}} = 1000$ and $N_{\text{test}} = 100$ sparse vectors $\boldsymbol{\beta}^{(n)}$ of size $D = 100$ from the spike and slab distribution with the truncated slab: each component $\beta_d^{(n)}$ is zero with the probability 0.8 or is sampled from the standard Gaussian distribution without interval $(-0.1, 0.1)$ with the probability 0.2. The design matrix $\mathbf{X}$ is random Gaussian. The observations $\mathbf{y}^{(n)}$ are generated as in (1) with the zero-mean Gaussian noise with the standard deviation 0.5. The shrinkage parameter is set to $\lambda = 0.1$. The algorithms are trained on the training data of size N_{train} and evaluated on the test data of size N_{test}.

In Figure 1a NMSE for different number of layers (or iterations) L is presented. The observation size is set to $K = 50$. BayesLISTA outperforms competitors. Figure 1b gives NMSE for different observation sizes K. The number of layers (iterations) is set as $L = 4$. In the previous experiment, Bayesian and classic LISTA show similar results with this number of layers. Figure 1b confirms this competitive behaviour between two LISTAs. ISTA and FISTA underperform the NNs.

3.2 Active learning

To demonstrate a potential scenario that can benefit from uncertainty estimates of BayesLISTA, we consider the active learning example [21]. The active learning area researches ways to select new training subsets to reduce the total number of required supervision. One of the popular approaches in active learning is uncertainty sampling, when the data with the least certain predictions is chosen for labelling. We use a variance of the spike and slab distributed prediction as a measure of uncertainty.

The MNIST dataset [22] is utilised. The dataset contains images of handwritten digits of size $28 \times 28 = 784$. The design matrix $\mathbf{X}$ is standard random Gaussian. Observations are generated as $\mathbf{y} = \mathbf{X}\boldsymbol{\beta}$, where $\boldsymbol{\beta} \in \mathbb{R}^{784}$ are flattened images. The shrinkage parameter λ is 0.1, the observation size K is 100.

We use the training data of size 50, the pool data of size 500, and the test data of size 100. The algorithm learns on the training data and it is evaluated on the test data. To actively collect a next data point from the pool, the algorithm is used to predict a point with the highest uncertainty. The selected point is moved from the pool to the training data and the algorithms learns on the updated training data. Overall, 10 pool additions are performed. After every addition the performance is

measured on the test data. We compare the active approach of adding new points from the pool with the random approach that picks a new data point from the pool at random. The procedure is repeated for 20 times.

Figure 1c demonstrates performance of the active and non-active methods of updates with BayesLISTA. The active approach with uncertainty sampling steadily demonstrates better results. This means the posterior distribution learnt by BayesLISTA is an adequate estimate of the true posterior.

Appendix C provides additional results on predictive performance on the MNIST data.

References

[1] Radford M Neal. *Bayesian learning for neural networks*. PhD thesis, University of Toronto, 1994.

[2] Yingzhen Li, José Miguel Hernández-Lobato, and Richard E Turner. Stochastic expectation propagation. In *Advances in Neural Information Processing Systems*, pages 2323–2331, 2015.

[3] Matthew D Hoffman, David M Blei, Chong Wang, and John Paisley. Stochastic variational inference. *The Journal of Machine Learning Research*, 14(1):1303–1347, 2013.

[4] Rajesh Ranganath, Linpeng Tang, Laurent Charlin, and David Blei. Deep exponential families. In *Artificial Intelligence and Statistics*, pages 762–771, 2015.

[5] Yarin Gal and Zoubin Ghahramani. Dropout as a Bayesian approximation: Representing model uncertainty in deep learning. In *International Conference on Machine Learning*, pages 1050–1059, 2016.

[6] José Miguel Hernández-Lobato and Ryan Adams. Probabilistic backpropagation for scalable learning of Bayesian neural networks. In *International Conference on Machine Learning*, pages 1861–1869, 2015.

[7] Daniel Soudry, Itay Hubara, and Ron Meir. Expectation backpropagation: Parameter-free training of multilayer neural networks with continuous or discrete weights. In *Advances in Neural Information Processing Systems*, pages 963–971, 2014.

[8] Patrick L McDermott and Christopher K Wikle. Bayesian recurrent neural network models for forecasting and quantifying uncertainty in spatial-temporal data. *arXiv preprint arXiv:1711.00636*, 2017.

[9] Robert Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 267–288, 1996.

[10] Michael E Tipping. Sparse Bayesian learning and the relevance vector machine. *Journal of machine learning research*, 1(Jun):211–244, 2001.

[11] Karol Gregor and Yann LeCun. Learning fast approximations of sparse coding. In *International Conference on Machine Learning*, pages 399–406, 2010.

[12] Pablo Sprechmann, Alexander M Bronstein, and Guillermo Sapiro. Learning efficient sparse and low rank models. *IEEE Transactions on pattern analysis and machine intelligence*, 37(9):1821–1833, 2015.

[13] Emmanuel J Candès and Michael B Wakin. An introduction to compressive sampling. *IEEE signal processing magazine*, 25(2):21–30, 2008.

[14] Julien Mairal, Francis Bach, Jean Ponce, et al. Sparse modeling for image and vision processing. *Foundations and Trends in Computer Graphics and Vision*, 8(2-3):85–283, 2014.

[15] Zhaowen Wang, Ding Liu, Jianchao Yang, Wei Han, and Thomas Huang. Deep networks for image super-resolution with sparse prior. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 370–378, 2015.

[16] Sylvain Baillet and Line Garnero. A Bayesian approach to introducing anatomo-functional priors in the EEG/MEG inverse problem. *IEEE transactions on Biomedical Engineering*, 44(5):374–385, 1997.

[17] Mainak Jas, Tom Dupré La Tour, Umut Simsekli, and Alexandre Gramfort. Learning the morphology of brain signals using alpha-stable convolutional sparse coding. In *Advances in Neural Information Processing Systems*, pages 1099–1108, 2017.

[18] Toby J Mitchell and John J Beauchamp. Bayesian variable selection in linear regression. *Journal of the American Statistical Association*, 83(404):1023–1032, 1988.

[19] Ingrid Daubechies, Michel Defrise, and Christine De Mol. An iterative thresholding algorithm for linear inverse problems with a sparsity constraint. *Communications on pure and applied mathematics*, 57(11): 1413–1457, 2004.

[20] Amir Beck and Marc Teboulle. A fast iterative shrinkage-thresholding algorithm with application to wavelet-based image deblurring. In *IEEE International Conference on Acoustics, Speech and Signal Processing, 2009. ICASSP 2009.*, pages 693–696. IEEE, 2009.

[21] Burr Settles. Active learning literature survey. Computer Sciences Technical Report 1648, University of Wisconsin–Madison, 2009.

[22] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[23] Thomas Minka. *A family of algorithms for approximate Bayesian inference*. PhD thesis, MIT, 2001.

A Appendix: Uncertainty propagation through soft-thresholding

This section describes modification of LISTA forward propagation (Algorithm 1) to include probability distributions of the random variables introduced in section 2.2.

Initialisation

At step 1 of LISTA (Algorithm 1) the matrix $\mathbf{W}$ consists of Gaussian-distributed components $w_{dk} \sim \mathcal{N}(m^w_{dk}, v^w_{dk})$, and $\mathbf{y}$ is a deterministic vector. Then the output $\mathbf{b}$ is a vector of Gaussian-distributed components $b_d \sim \mathcal{N}(m^b_d, v^b_d)$, where $m^b_d = \sum_{k=1}^K y_k m^w_{dk}$, and $v^b_d = \sum_{k=1}^K y_k^2 v^w_{dk}$.

At step 2 of LISTA (Algorithm 1) the Gaussian vector $\mathbf{b}$ is taken as an input of the soft-thresholding function. When a Gaussian random variable $x \sim \mathcal{N}(x; m, v)$ is propagated through the soft-thresholding function $x^* = h_\lambda(x)$, the probability mass of the resulting random variable x^* is split into two parts. The values of x from the interval $[-\lambda, \lambda]$ are converted to 0 by the soft-thresholding operator. Therefore, the probability mass of the original distribution that lies in $[-\lambda, \lambda]$ is squeezed into the probability of x^* being zero. The values of x from outside of the $[-\lambda, \lambda]$ interval are shifted towards 0. The distribution of $x^* \neq 0$ then represents the tails of the original Gaussian distribution. The distribution of x^* can be then parametrised by the probability of being zero, ω^*, the mean m^* and the variance v^* of the truncated Gaussian distribution. Therefore, we approximate the distribution of $\widehat{\boldsymbol{\beta}}_0$ at step 2 with a spike and slab distribution with parameters: the spike probability ω^*, the slab mean m^* and variance v^*.

Main layers

At step 4 of LISTA (Algorithm 1) the vector $\mathbf{b}$ and matrix $\mathbf{S}$ consist of Gaussian components: $b_d \sim \mathcal{N}(m^b_d, v^b_d)$, $s_{d'd''} \sim \mathcal{N}(m^s_{d'd''}, v^s_{d'd''})$, and $\widehat{\boldsymbol{\beta}}_{l-1}$ is a vector of the spike and slab random variables: $[\widehat{\beta}_{l-1}]_d \sim \omega_d \delta_0 + (1-\omega_d)\mathcal{N}(m_d, v_d)$.

It can be shown that the expected value and variance of a spike and slab distributed variable ξ with the probability of spike ω, the slab mean m and slab variance v are:

$$\mathbb{E}\xi = (1-\omega)m, \qquad \operatorname{Var}\xi = (1-\omega)(v + \omega m^2). \tag{8}$$

It can also be shown that if components of the matrix $\mathbf{S}$ and vector $\widehat{\boldsymbol{\beta}}_{l-1}$ are mutually independent then the components $[\mathbf{e}_l]_d$ of their product $\mathbf{e}_l = \mathbf{S}\widehat{\boldsymbol{\beta}}_{l-1}$ have the marginal mean and variances:

$$m^e_d \overset{\text{def}}{=} \mathbb{E}[\mathbf{e}_l]_d = \sum_{d'=1}^D m^s_{dd'}(1-\omega_{d'})m_{d'}, \tag{9a}$$

$$v^e_d \overset{\text{def}}{=} \operatorname{Var}[\mathbf{e}_l]_d = \sum_{d'=1}^D [(m^s_{dd'})^2(1-\omega_{d'})^2 v_{d'} + (1-\omega_{d'})^2 (m_{d'})^2 v^s_{dd'} + v^s_{dd'}(1-\omega_{d'})^2 v_{d'}]. \tag{9b}$$

According to the Central Limit Theorem $[\mathbf{e}_l]_d$ can be approximated as a Gaussian-distributed variable when D is sufficiently large. The parameters of this Gaussian distribution are the marginal mean and variance given in (9).

The output $\mathbf{c}_l$ at step 4 is then represented as a sum of two Gaussian-distributed vectors: $\mathbf{b}$ and $\mathbf{e}_l$, i.e. it is a Gaussian-distributed vector with components $c_d \sim \mathcal{N}(m_d^c, v_d^c)$, where $m_d^c = m_d^b + m_d^e$ and $v_d^c = v_d^b + v_d^e$.

Then $\widehat{\boldsymbol{\beta}}_l$ at step 5 of LISTA (Algorithm 1) is the result of soft-thresholding of a Gaussian variable, which is approximated with the spike and slab distribution, similar to step 2 (section A). Thus, all the steps of BayesLISTA are covered and distributions for outputs of these steps are derived.

B Appendix: Backpropagation

The exact intractable posterior (7) is approximated with a factorised distribution

$$q(\mathbf{W}, \mathbf{S}, \gamma, \eta) = \prod_{d=1}^{D} \prod_{k=1}^{K} \mathcal{N}(w_{dk}; m_{dk}^w, v_{dk}^w) \prod_{d'=1}^{D} \prod_{d''=1}^{D} \mathcal{N}(s_{d'd''}; m_{d'd''}^s, v_{d'd''}^s) \times \text{Gam}(\gamma; a^\gamma, b^\gamma)\text{Gam}(\eta; a^\eta, b^\eta) \tag{10}$$

Parameters of approximating distributions are updated with the assumed density filtering (ADF) and expectation propagation (EP) algorithms derived on the derivatives of the logarithm of a normalisation constant (based on [6]). ADF iteratively incorporates factors from the true posterior p in (7) into the factorised approximating distribution q in (10), whereas EP iteratively replaces factors in q by factors from p.

When a factor from p is incorporated into q, q has the form $q(a) = Z^{-1} f(a)\mathcal{N}(a; m, v)$ as a function of weights $\mathbf{W}$ and $\mathbf{S}$, where Z is the normalisation constant and $f(a)$ is an arbitrary function, $a \in \{w_{dk}, s_{d'd''}\}$. New parameters of the Gaussian distribution for a can be computed as [23]

$$m := m + v\frac{\partial \log Z}{\partial m},\; v := v - v^2\left[\left(\frac{\partial \log Z}{\partial m}\right)^2 - 2\frac{\partial \log Z}{\partial v}\right] \tag{11}$$

Then for new values of $\mathbf{W}$ and $\mathbf{S}$ derivatives of the logarithm of Z are required when the factor of p is incorporated in q.

With the likelihood factors (5) of p the ADF approach is employed and they are iteratively incorporated into q. The normalisation constant of q with the likelihood term for the data point n incorporated is (let z_d denote (to simplify notation the superscript (n) is omitted)

$$Z = \int \prod_{d=1}^{D} \mathcal{N}(\beta_d; [f(\mathbf{y}; \mathbf{S}, \mathbf{W}, \lambda)]_d, \gamma^{-1}) q(\mathbf{W}, \mathbf{S}, \gamma, \eta) \mathrm{d}\mathbf{W}\mathrm{d}\mathbf{S}\mathrm{d}\gamma\mathrm{d}\eta \tag{12}$$

Assuming the spike and slab distribution for $\widehat{\boldsymbol{\beta}}$, the normalisation constant can be approximated as

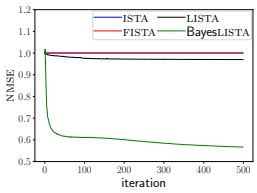
$$Z \approx \prod_{d=1}^{D} \left[\omega_d^{\widehat{\boldsymbol{\beta}}} \mathcal{T}\left(\beta_d; 0, \beta^\gamma/\alpha^\gamma, 2\alpha^\gamma\right) + \left(1 - \omega_d^{\widehat{\boldsymbol{\beta}}}\right) \mathcal{N}\left(\beta_d; m_d^{\widehat{\boldsymbol{\beta}}}, \beta^\gamma/(\alpha^\gamma - 1) + v_d^{\widehat{\boldsymbol{\beta}}}\right)\right], \tag{13}$$

where $\{\omega_d^{\widehat{\boldsymbol{\beta}}}, m_d^{\widehat{\boldsymbol{\beta}}}, v_d^{\widehat{\boldsymbol{\beta}}}\}$ are the parameters of the spike and slab distribution for $[\widehat{\boldsymbol{\beta}}]_d$. Parameters of q are then updated with the derivatives of Z according to (11).

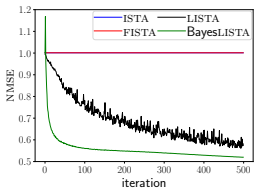
Prior factors (3) and (6) from p are incorporated into q with the EP algorithm [6], i.e. they replace the corresponding approximating factors from q, and then q is updated to minimise the Kullback–Leibler divergence.

C Appendix: Predictive performance on MNIST data

In this experiment, the methods are evaluated on the MNIST dataset in terms of predictive performance. We use 100 images for training and 100 for test.



(a) MNIST for $K = 100$



(b) MNIST for $K = 250$

Figure 2: NMSE results on the MNIST data for increasing number of iterations with the observation size $K = 100$ (a) and $K = 250$ (b)

Figures 2a and 2b present NMSE with observation sizes 100 and 250. The experiment with $K = 100$ presents severe conditions for the algorithms: the limited size of the training dataset combined with the small dimensionality of observations. BayesLISTA is able to learn under these conditions, significantly outperforming LISTA. Under better conditions of the second experiment with $K = 250$, both NNs converge to the similar results. However, BayesLISTA demonstrates a remarkably better convergence rate. ISTA and FISTA are unable to perform well in these experiments.