Distributed Bayesian Learning with Stochastic Natural-gradient Expectation Propagation

Leonard Hasenclever University of Oxford
Stefan Webb University of Oxford
Thibaut Lienart University of Oxford
Sebastian Vollmer University of Oxford

Balaji Lakshminarayanan Google DeepMind
Charles Blundell Google DeepMind
Yee Whye Teh University of Oxford

Abstract

This paper makes two contributions to Bayesian machine learning algorithms. Firstly, we propose stochastic natural gradient expectation propagation (SNEP), a novel black box variational algorithm that is an alternative to expectation propagation (EP). In contrast to EP which has no guarantee of convergence, SNEP can be shown to be convergent, even when using Monte Carlo moment estimates. Secondly, we propose a novel architecture for distributed Bayesian learning which we call the posterior server, implementing a distributed asynchronous version of SNEP, which allows scalable and robust Bayesian learning in cases where a dataset is stored in a distributed manner across a cluster. An independent Monte Carlo sampler is run on each compute node which targets an approximation to the global posterior distribution given all data across the whole cluster. We demonstrate SNEP and the posterior server on distributed Bayesian logistic regression.

1 Introduction

Bayesian methods provide a principled way to reason about uncertainty and prevent overfitting. However, Bayesian methods are generally more computationally intensive than optimisation-based ones and, as a result, most of recent successes of large-scale machine learning are driven by optimization based methods. Hence there is a need for scalable Bayesian machine learning.

Since the posterior distribution is generally intractable in all but the simplest models , approximate methods have to be applied. In this paper we consider *distributed* Bayesian learning. There is a lot of work on embarrassingly parallel MCMC [HG05, SBB$^+$13, WD13, NWX14] which distributes data across a cluster, runs independent MCMC samplers on each worker and combines samples across the cluster only at the end to reduce network communication costs. This approach has the disadvantage that it combines N samples from local posteriors into only one approximate sample from the global posterior and introduces a large approximation error. In particular the separate sampler explore parts of the parameter space that are unlikely given data on other workers thus wasting resources.

Our work builds upon prior work on using expectation propagation (EP) for performing distributed Bayesian learning [XLT$^+$14, GVJ$^+$14]. Xu et al. (2014) called this approach sampling via moment sharing (SMS). In this framework, a dataset is partitioned into disjoint subsets with each subset stored on a worker node in a cluster. Learning is performed at each worker based on the data subset there using MCMC sampling. As opposed to embarrassingly parallel MCMC methods which only communicate the samples to the master at the end of learning, EP is used to communicate messages (infrequently) across the cluster. These messages coordinate the samplers such that the target distributions of all samplers on all workers share certain moments, e.g. means and variances, hence the name . At convergence, it can be shown that the target distributions of the samplers also

30th Conference on Neural Information Processing Systems (NIPS 2016), Barcelona, Spain.

share moments with the EP approximation to the global posterior distribution given all data. Hence the target distributions on the workers can themselves be treated as approximations to the global posterior and samples obtained on the workers can be treated as approximate samples from the posterior. While SMS works well on simpler models like Bayesian logistic regression, we have found that it did not work for more complex, high-dimensional, and non-convex models like Bayesian deep neural networks due to the non-convergence of EP and the fact that we use noisy moment estimates.

Our first contribution is thus the development of stochastic natural-gradient EP (SNEP), an alternative algorithm to power EP (a generalisation of EP) [Min04] which optimises the same variational objective function. SNEP is a double-loop algorithm with convergence guarantees. The inner loop is a stochastic natural-gradient descent algorithm which tolerates estimation noise, so that SNEP is convergent even with moments estimated using MCMC samplers. Our derivation of SNEP improves upon the derivation of the convergent EP algorithm of [HZ02] in that ours works for a more general class of models, we make explicit the underlying variational objective function that is being optimised, and ours uses a natural-gradient descent algorithm [AN01].

Building upon the development of SNEP, our second contribution is a distributed Bayesian learning architecture which we call the posterior server. In analogy to the parameter server [AAG+12] which maintains and serves the parameter vector to a cluster of workers, the posterior server maintains and serves (an approximation to) the posterior distribution. Each worker has a subset of data and maintains a tractable approximation of the likelihood and a cavity distribution which is effectively a conditional distribution over the parameters given all data on other workers. An MCMC sampler targets the normalised product of the cavity distribution and the (true) likelihood, and forms a stochastic estimate of the required moments, which is in turn used to update the likelihood approximation using stochastic natural-gradient descent. Each worker communicates with the posterior server asynchronously and in a non-blocking manner, sending the current likelihood approximation and receiving the new cavity distribution. This communication protocol makes more efficient use of computational resources on workers than SMS, which requires either synchronous or blocking asynchronous protocols. In the following we will briefly sketch out the derivation of SNEP and discuss related works in section 2, present experimental results in section 3 and conclude in section 4.

2 Sketch derivation of SNEP

Following the formalism of Wainwright and Jordan [WJ08] for EP let us consider an exponential family with density:

$$p_\theta(x) = \exp\left(\theta^\top s(x) - A(\theta)\right), \quad \text{where} \quad A(\theta) = \log \int \exp\left(\theta^\top s(x)\right) dx. \tag{1}$$

where $s(x)$ is the sufficient statistics function, θ is the natural parameter and $A(\theta)$ is the log-partition function. Let $\Theta = \{\theta : A(\theta) < \infty\}$ be the natural domain. Associated with this exponential family is a mean parameter $\mu = \mathbb{E}_{p_\theta}[s(x)]$ and a mean domain $\mathcal{M} = \{\mu : \exists p \; s.t. \; \mathbb{E}_p[s(X)] = \mu\}$. Under regularity conditions, the mapping $\theta \mapsto \nabla A(\theta)$ is one-to-one and onto the interior of $\mathcal{M}$, and maps θ to the mean parameter, $\mu(\theta) = \nabla A(\theta)$. It can be shown that A is a convex function and Θ and $\mathcal{M}$ are convex sets. The convex conjugate of $A(\theta)$ is,

$$A^*(\mu) := \sup_{\theta \in \Theta} \theta^\top \mu - A(\theta), \text{ and conversely } A(\theta) = \sup_{\mu \in \mathcal{M}} \theta^\top \mu - A^*(\mu) \tag{2}$$

Let ℓ_i denote the loglikelihood of the part of the data on worker i. In general, the likelihood functions are intractable and approximations are necessary. Our works extends *power expectation propagation* (power EP) [Min04]. To start with, we may trivially formulate the target posterior distribution as an *extended exponential family distribution* with sufficient statistics $\tilde{s}(x) := [s(x), \ell_1(x), \ldots, \ell_n(x)]$ and natural parameters $\tilde{\theta} := [\theta_0, \mathbf{1}_n]$ where $\mathbf{1}_n$ is a vector of 1's of length n. Power EP is formulated in terms of simpler exponential families, known as *locally extended exponential families*. For each i, let the ith locally extended exponential family be associated with the sufficient statistics function $s_i(x) := [s(x), \ell_i(x)]$. Let Θ_i, $\mathcal{M}_i$, A_i, A_i^* be the associated (local) natural domain, mean domain, log partition function and negative entropy respectively. A distribution in this locally extended exponential family with natural parameter $[\theta_i, \eta_i] \in \Theta$ has the form

$$p_{\theta_i, \eta_i}(x) = \exp\left(\theta_i^\top s(x) + \eta_i \ell_i(x) - A_i(\theta, \eta_i)\right), \tag{3}$$

which can be seen as the posterior distribution where the likelihood from all other workers has been replaced by an exponential family approximation. In the formulation of Wainwright and Jordan [WJ08] the Power EP variational problem can then be written as:

$$\begin{aligned} \max_{\substack{\mu_0 \in \mathcal{M} \\ [[\mu_i,\nu_i]\in\mathcal{M}_i]_{i=1}^n}} \quad & \theta_0^\top \mu_0 + \sum_{i=1}^n 1 \cdot \nu_i - A^*(\mu_0) - \sum_{i=1}^n \beta_i (A_i^*(\mu_i,\nu_i) - A^*(\mu_i)) \\ \text{subject to} \quad & \mu_0 = \mu_i \quad \text{for } i = 1,\dots,n \end{aligned} \tag{4}$$

The Power EP updates can be derived as a fixed point scheme to solve this optimization problem. SNEP instead uses a modified variational objective with the same optima and solves the dual problem using a stochastic approximation algorithm [RM51]. To this end we introduce an auxiliary natural parameter vector $\theta_i' \in \Theta$ for each i, and introduce a term $-\sum_{i=1}^n \mathrm{KL}(\mu_i \| \theta_i')$. This results in a lower bound to the origin problem. Maximizing over θ_i' while keeping the other variables fixed recovers the original problem. Hence we consider maximizing over θ_i' in an outer loop while performing the other optimizations in an inner loop. Introducing Lagrange multipliers to enforce the equality constraints, we have,

$$\max_{[\theta_i'\in\Theta]_{i=1}^n} \max_{\substack{\mu_0 \in \mathcal{M} \\ [[\mu_i,\nu_i]\in\mathcal{M}_i]_{i=1}^n}} \min_{[\lambda_i]_{i=1}^n} \theta_0^\top \mu_0 - A^*(\mu_0) + \sum_{i=1}^n \left(\nu_i - \lambda_i^\top(\mu_i - \mu_0) - \beta_i \left(A_i^*(\mu_i,\nu_i) - \mu_i^\top \theta_i' + A(\theta_i')\right)\right)$$

Noticing that the Lagrangian is concave in $\mu_0, [\mu_i,\nu_i]_{i=1}^n$ and that Slater's condition holds, the duality gap is zero and we have

$$\max_{[\theta_i'\in\Theta_i]_{i=1}^n} \min_{[\lambda_i]_{i=1}^n} \max_{\substack{\mu_0 \in \mathcal{M} \\ [[\mu_i,\nu_i]\in\mathcal{M}_i]_{i=1}^n}} \theta_0^\top \mu_0 - A^*(\mu_0) + \sum_{i=1}^n \left(\nu_i - \lambda_i^\top(\mu_i - \mu_0) - \beta_i \left(A_i^*(\mu_i,\nu_i) - \mu_i^\top \theta_i' + A(\theta_i')\right)\right)$$

Finally, maximizing over $\mu_0, [\mu_i,\nu_i]_{i=1}^n$, we have the equivalent dual problem, whose objective function we will denote with $L([\theta_j', \lambda_j]_{j=1}^n)$,

$$\max_{[\theta_i'\in\Theta]_{i=1}^n} \min_{[\lambda_i]_{i=1}^n} \underbrace{A\left(\theta_0 + \sum_{i=1}^n \lambda_i\right) + \sum_{i=1}^n \beta_i \left(A_i\left(\theta_i' - \beta_i^{-1}\lambda_i, \beta_i^{-1}\right) - A(\theta_i')\right)}_{=:L([\theta_j',\lambda_j]_{j=1}^n)} \tag{5}$$

Instead of using a fixed point iteration as in EP we will use a stochastic gradient descent algorithm in the inner loop to better deal with the noise in Monte Carlo moment estimates. Instead of updating the natural parameters λ_i of the i-th likelihood approximation directly we can reparametrise in terms of the corresponding mean parameter which allows us to use a natural gradient [AN01] or, equivalently, mirror descent [BT03, RM15] at no additional computational cost instead (see [HWL+16] for details). Reparameterising with $\lambda_i = \nabla A^*(\gamma_i)$, the resulting update is

$$\gamma_i^{(t+1)} = \gamma_i^{(t)} + \epsilon_t \left(\nabla_{\theta_i} A_i \left(\theta_i' - \beta_i^{-1}\lambda_i^{(t)}, \beta_i^{-1}\right) - \nabla A\left(\theta_0 + \sum_{j=1}^n \lambda_j^{(t)}\right)\right) \tag{6}$$

where ϵ_t is the step size at iteration t and we used the notation $\nabla_{\theta_i} A_i$ for the partial derivative of $A_i(\cdot,\cdot)$ with respect to its first argument. The first term of this update is intractable and has to be estimated using MCMC. In practice we found that using just one step of an MCMC sampler to estimate this mean parameter was sufficient for fast convergence. These updates can be performed in series or parallel fashion. In our distributed Bayesian learning setting they are performed in an asynchronous distributed fashion (see appendix A for the full algorithm).

3 Experiments

In this section we compare SNEP against *Sampling via Moment Sharing* (SMS) [XLT+14], a related algorithm for distributed Bayesian learning described above, when applied to Bayesian logistic regression with simulated data. Following the SMS paper [XLT+14], generating a dataset $\mathcal{D} = \{(z_c, y_c)\}_{c=1}^N$ with covariates $z_c \in \mathbb{R}^d$ and response $y_c \in \{0,1\}$. We used a Gaussian prior

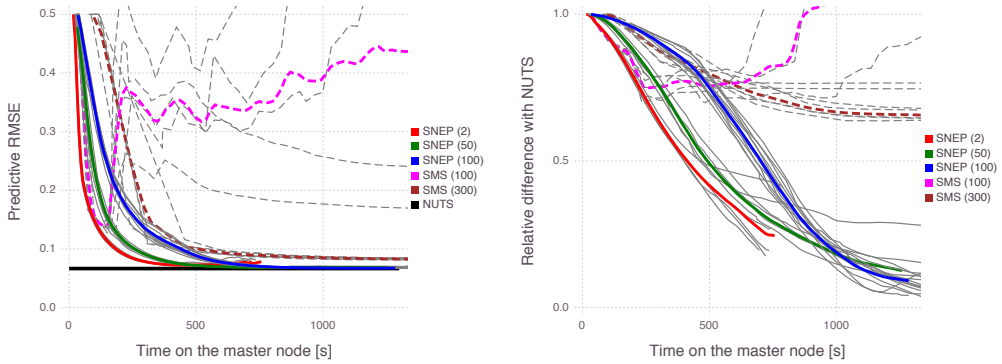


Figure 1: (left) Comparison of the predictive RMSE obtained with SMS (dashed-lines) and SNEP (full lines) versus wallclock time when varying the number of MCMC samples per iteration (numbers reported in the legend). When too few samples are used for the moment estimates, SMS becomes unstable whilst SNEP does not suffer from this. SNEP is also consistently faster and more accurate than SMS. The accuracy obtained with the posterior mean estimated with a long run of Stan/NUTS is also displayed as comparison (horizontal line). (right) Relative difference between the posterior means estimated using Stan/NUTS and using SMS (dashed lines) or SNEP (full lines). Each coloured line corresponds to an average over several runs (light grey).

$p_0(x) = \mathcal{N}(x; 0_d, 10I_d)$ on the weights x and the aim is to construct an approximation to the posterior $p(x \mid \mathcal{D})$. We generated $N = 50000$ data points with $d = 50$ using iid draws for the covariates, $z_c \sim \mathcal{N}(\mu, \Sigma)$ where $\mu \in [0, 1]^d$ and $\Sigma = PP^\top$ with $P \in [-1, 1]^{d \times d}$, with entries drawn uniformly at random for both P and μ. The generating weight vector x^* is drawn from the prior $\mathcal{N}(x; 0_d, 10I_d)$. The labels y_c are then sampled according to the model.

Both algorithms were run with three workers each with one third of the data. SNEP is run with 1 inner loop iteration per synchronisation with the master (for the purpose of comparing against SMS). Varying numbers of MCMC samples per inner loop iteration were used for both algorithms, to investigate the effect of Monte Carlo noise on the performances of the algorithms (low number of samples meaning high noise and both lower performance and lower computational cost). The damping for SMS and the learning rate for SNEP were tuned for best performances. As the base exponential family, we used a full-covariance Gaussian. We compared the predictive RMSE $\sqrt{\sum_c |\hat{p}_c - y_c|^2 / N}$. We also compared the relative difference between the estimated posterior mean and that estimated from a long run of the No-U-Turn sampler [HG14] in Stan [CGH$^+$ss] (see figure 1). It can be observed that SNEP is more robust and better performing than SMS and requires many fewer samples especially in high-dimensional settings.

We also applied SNEP to Bayesian deep learning where SNEP is competitive with the best distributed algorithms [ZCL15]. Some of these experiments can be found in the appendix. Further experiments in [HWL$^+$16] show that SNEP is robust to the length of the communication intervals and other hyperparameters.

4 Conclusion

We introduced SNEP, a distributed algorithm for Bayesian learning in complex models and presented experiments showing its performance in Bayesian logistic regression. SNEP introduces auxiliary variables into the Power EP problem and uses a double loop algorithm to solve the resulting optimization problem. In practice we found that only a few inner loop iterations per outer loop iteration are needed and only one MCMC step to estimate the mean parameters for the inner loop iteration was sufficient for fast convergence. Our experimental results are encouraging and show that it is possible to deploy approximate Bayesian inference techniques in a distributed setting. This is important for large-scale machine learning and could also prove useful in situations where data is distributed and not shared because of privacy concerns. Further research is needed to fully understand its convergence properties, dependence on hyperparameters and explore its performance on larger models.

References

[AAG+12] A. Ahmed, M. Aly, J. Gonzalez, S. Narayanamurthy, and A. J. Smola. Scalable inference in latent variable models. In *Proceedings of the ACM International Conference on Web Search and Data Mining (WSDM)*, 2012.

[AN01] S. Amari and H. Nagaoka. *Methods of Information Geometry*. American Mathematical Society, 2001.

[BT03] A. Beck and M. Teboulle. Mirror descent and nonlinear projected subgradient methods for convex optimization. *Operations Research Letters*, 31(3):167–175, 2003.

[CGH+ss] B. Carpenter, A. Gelman, M. Hoffman, D. Lee, B. Goodrich, M. Betancourt, M. A. Brubaker, J. Guo, P. Li, and A. Riddell. Stan: A probabilistic programming language. *Journal of Statistical Software*, in press.

[DCM+12] Jeffrey Dean, Greg Corrado, Rajat Monga, Kai Chen, Matthieu Devin, Mark Mao, Andrew Senior, Paul Tucker, Ke Yang, Quoc V Le, and Andrew Y. Ng. Large scale distributed deep networks. In *Advances in Neural Information Processing Systems*, pages 1223–1231, 2012.

[GVJ+14] A. Gelman, A. Vehtari, P. Jylänki, C. Robert, N. Chopin, and J. P. Cunningham. Expectation propagation as a way of life. *ArXiv:1412.4869*, 2014.

[HG05] Z. Huang and A. Gelman. Sampling for Bayesian computation with large datasets. Technical report, Department of Statistics, Columbia University, 2005.

[HG14] M. D. Hoffman and A. Gelman. The no-U-turn sampler: Adaptively setting path lengths in Hamiltonian Monte Carlo. *Journal of Machine Learning Research*, 15, 2014.

[HWL+16] Leonard Hasenclever, Stefan Webb, Thibaut Lienart, Sebastian Vollmer, Balaji Lakshminarayanan, Charles Blundell, and Yee Whye Teh. Distributed Bayesian Learning with Stochastic Natural-gradient Expectation Propagation and the Posterior Server. *arxiv preprint arxiv:1512.09327*, 2016.

[HZ02] T. Heskes and O. Zoeter. Expectation propagation for approximate inference in dynamic Bayesian networks. In *Proceedings of the Conference on Uncertainty in Artificial Intelligence*, volume 18, 2002.

[KB15] D. P. Kingma and J. L. Ba. Adam: A method for stochastic optimization. In *Proceedings of the International Conference on Learning Representations*, 2015.

[LCCC16] Chunyuan Li, Changyou Chen, David Carlson, and Lawrence Carin. Preconditioned Stochastic Gradient Langevin Dynamics for Deep Neural Networks. *AAAI*, dec 2016.

[Min04] T. Minka. Power EP. Technical report, Microsoft Research, 2004.

[NVL+15] Arvind Neelakantan, Luke Vilnis, Quoc V Le, Ilya Sutskever, Lukasz Kaiser, Karol Kurach, and James Martens. Adding gradient noise improves learning for very deep networks. *arXiv preprint arXiv:1511.06807*, 2015.

[NWX14] W. Neiswanger, C. Wang, and E. P. Xing. Asymptotically exact, embarrassingly parallel MCMC. In *Proceedings of the International Conference on Uncertainty in Artificial Intelligence*, 2014.

[RM51] H. Robbins and S. Monro. A stochastic approximation method. *Annals of Mathematical Statistics*, 22(3):400–407, 1951.

[RM15] G. Raskutti and S. Mukherjee. The information geometry of mirror descent. *IEEE Transactions on Information Theory*, 61:1451–1457, 2015.

[SBB+13] S. L. Scott, A. W. Blocker, F. V. Bonassi, H. Chipman, E. George, and R. McCulloch. Bayes and big data: the consensus Monte Carlo algorithm. In *Bayes 250*, 2013.

[WD13] X. Wang and D. B. Dunson. Parallelizing MCMC via Weierstrass sampler, 2013.

[WJ08] M. J. Wainwright and M. I. Jordan. Graphical models, exponential families, and variational inference. *Foundations and Trends in Machine Learning*, 1(1-2):1–305, 2008.

[WT11] M. Welling and Y. W. Teh. Bayesian learning via stochastic gradient Langevin dynamics. In *Proceedings of the International Conference on Machine Learning*, 2011.

[XLT+14] M. Xu, B. Lakshminarayanan, Y. W. Teh, J. Zhu, and B. Zhang. Distributed Bayesian posterior sampling via moment sharing. In *Advances in Neural Information Processing Systems*, 2014.

[ZCL15] S. Zhang, A. Choromanska, and Y. LeCun. Deep learning with elastic averaging SGD. *Advances in Neural Information Processing Systems*, 2015.

A Distributed SNEP algorithm

Algorithm 1 Posterior Server: Distributed Bayesian Learning via SNEP

for each compute node $i = 1, \ldots, n$ **asynchronously do**
  let $\gamma_i^{(1)}$ be the initial mean parameter of local likelihood approximation.
  let $\lambda_i^{\text{old}} := \lambda_i^{(1)} := \nabla A^*(\gamma_i^{(1)})$ be the initial natural parameter of local likelihood approximation.
  let $\theta_{-i} := \theta_0 + \sum_{j \neq i} \lambda_j^{(1)}$ be the initial natural parameter of cavity distribution
  let $\theta_i' := \theta_{-i} + \lambda_i^{(1)}$ be the initial auxiliary parameter.
  let $x_i^{(1)} \sim p_{\theta_{-i} + \lambda_i^{(1)}}$ be the initial state of MCMC sampler.
  for $t = 1, 2, \ldots$ until convergence **do**
    update local state via MCMC targeting the tilted distribution:

$$x_i^{(t+1)} \sim \mathcal{K}_i \left(\cdot \mid x_i^{(t)}; \theta_i' - \beta_i^{-1} \lambda_i^{(t)}, \beta_i^{-1} \right)$$

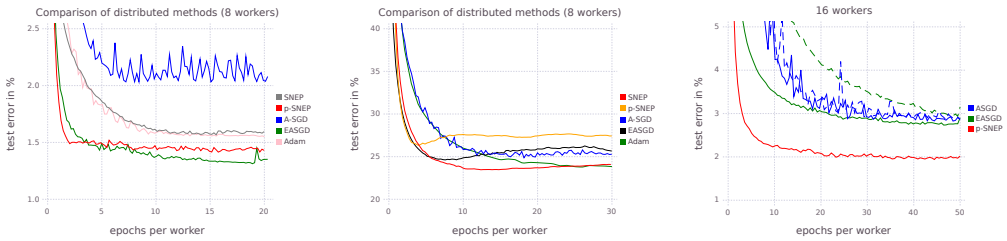
    update local likelihood approximation:

$$\gamma_i^{(t+1)} := \gamma_i^{(t)} + \epsilon_t \left(s(x_i^{(t+1)}) - \nabla A \left(\theta_{-i} + \lambda_i^{(t)} \right) \right)$$
$$\lambda_i^{(t+1)} := \nabla A^*(\gamma_i^{(t+1)})$$

    every N_{outer} iterations **do**: update auxiliary parameter:

$$\theta_i' := \theta_{-i} + \lambda_i^{(t)}$$

    every N_{sync} iterations **asynchronously do:** communicate with posterior server:
      let $\Delta_i := \lambda_i^{(t)} - \lambda_i^{\text{old}}$.
      update $\lambda_i^{\text{old}} := \lambda_i^{(t)}$.
      send Δ_i to posterior server.
      receive $\theta_{\text{posterior}}$ from posterior server.
      update $\theta_{-i} := \theta_{\text{posterior}} - \lambda_i^{\text{old}}$.
  end for
end for
for the posterior server **do**
  let $\theta_{\text{posterior}} := \theta_0 + \sum_{j=1}^n \lambda_i^{(1)}$ be the initial natural parameter of the posterior approximation.
  maintain a queue of messages from workers.
  for each message Δ_i received from some worker i **do**
    update $\theta_{\text{posterior}} := \theta_{\text{posterior}} + \Delta_i$.
    send $\theta_{\text{posterior}}$ to worker i.
  end for
end for



(a) (b) (c)

Figure 2: Comparing (p-)SNEP to non-Bayesian distributed learning algorithms with: (a) a shallow dense architecture on MNIST; (b) a small CNN on CIFAR-10; (b) a deep dense architecture with twenty hidden layers on MNIST; In (c), the dashed lines for A-SGD and EASGD indicate that a prelearning phase was performed, whilst for p-SNEP no prelearning was found necessary.

B Experiments with Bayesian Neural Networks

In this section we report experimental results applying SNEP and the posterior server to distributed Bayesian learning of neural networks. We will compare our algorithm to Adam [KB15], a state-of-the-art stochastic gradient descent (SGD) algorithm with access to the whole dataset on a single computer, as well as several state-of-the-art distributed SGD algorithms: asynchronous SGD (A-SGD) [DCM$^+$12] and elastic averaging SGD (EASGD) [ZCL15]. In our experiments we used stochastic gradient Langevin dynamics [WT11] with an preconditioning scheme reminiscent of Adam as the MCMC sampler. The preconditioning scheme is the same as the one proposed by [LCCC16] except for the addition of a debiasing reminiscent of Adam.

In a first set of experiments we applied our algorithm to the MNIST data set of handwritten digits using a deep neural network with two hidden layers of 500 and 300 hidden units (see Figure 2(a)). There are two versions of our algorithm, SNEP and pSNEP, corresponding to slightly different objectives. As can be seen in the figure, pSNEP is competitive with EASGD in this experiment. Both algorithms outperform A-SGD. Figure 2(b) shows the same comparison on CIFAR10 with a small CNN. Here SNEP performs better than pSNEP and converges faster than all other algorithms.

In another set of experiments, we compared p-SNEP to a deep feedforward network with twenty hidden layers of dimension fifty (see Figure 2(c)). [NVL$^+$15] recently used this architecture to demonstrate the advantages of adding noise to standard SGD. We found that while adding noise to SGD, A-SGD, and EASGD did help some runs escape suboptimal solutions, it did not allow any of these methods to obtain a solution like that found by p-SNEP with extra percent of accuracy. Thus, this suggests that the benefits to learning with SNEP cannot entirely be put down to the addition of noise. Further experiments in [HWL$^+$16] show that SNEP is robust to the length of the communication intervals.