SPEEDING UP CYCLIC UPDATE SCHEMES BY PATTERN SEARCHES

Antti Honkela

Helsinki University of Technology, Neural Networks Research Centre
P.O. Box 5400, FIN-02015 HUT, Espoo, Finland
`Antti.Honkela@hut.fi` `http://www.cis.hut.fi/projects/ica/bayes/`

ABSTRACT

A popular strategy for dealing with large parameter estimation problems is to split the problem into manageable subproblems and solve them cyclically one by one until convergence. We address a well-known problem with this strategy, namely slow convergence under low noise. We propose using so called pattern searches which consist of a parameter-wise update phase followed by a line search. The search direction of the line search is computed by combining the individual updates of all subproblems. The approach can be used to accelerate learning of several methods proposed in the literature without the need for large algorithmic modifications such as evaluation of global gradients. The proposed modification is shown to reduce the convergence time in a realistic independent component analysis (ICA) problem by more than 85 %.

1. INTRODUCTION

Many existing learning methods use a simple cyclic scheme for updating the parameters: they take one parameter or a group of parameters at a time and optimize it while the others are fixed to their present values. When the process is repeated for different sets of parameters, the values eventually converge to an optimum. In particular, this strategy is utilized in the variational Bayesian techniques which decouple the difficult problem of describing the posterior probability density of the model parameters into many smaller tractable problems [1, 2, 3, 4, 5, 6, 7, 8, 9]. Theoretical justification for the method as an optimization algorithm and some different applications can be found in [10].

It is well known that in the case of low noise, the posterior dependences are stronger and consequently the cyclic update procedure is slow. It is difficult to optimize the values of dependent variables one at a time since each value can only be changed very little if the other parameters stay constant.

As an example, let us consider a simple linear model $\mathbf{x} = \mathbf{A}\mathbf{s}$, where $\mathbf{x}$ denotes the observation vector, $\mathbf{s}$ the source vector and $\mathbf{A}$ the mixing matrix. This is the generative model for instance in linear independent component analysis (ICA) [11]. ICA estimation means looking for a suitable rotation for the mixing matrix $\mathbf{A}$ such that the components of $\mathbf{s}$ would become as independent as possible. In this problem, the mixing matrix $\mathbf{A}$ and the source vectors $\mathbf{s}$ are intimately tied. Rotating one needs to be compensated by a suitable rotation of the other. If the mixing matrix and the sources are updated separately, a very large number of steps may be required to find the correct rotation as both parameters can only be changed very little at each step.

The effect of noise level to the set of feasible solutions of a simplified one-dimensional version of this linear model is illustrated in Fig. 1. At moderate noise levels (upper subfigure) the maximum of the posterior distribution is shallow and the basic cyclic iteration scheme converges quickly. When the noise level decreases (lower subfigure), the feasible region becomes much narrower and convergence of cyclic iteration slows down considerably.

The pattern search method we propose is a simple extension to the standard update strategy and requires no problem-specific modifications. Contrary to other popular optimization methods such as the conjugate gradient method, the pattern search method does not need the derivatives of the cost function. The algorithm is presented in detail in Section 2. In Section 3, the method is used to speed up two experiments using variational Bayesian learning. The paper concludes with discussion and suggestions for future work in Section 4.

2. THE ALGORITHM

We propose speeding up convergence of cyclic update schemes by applying the the idea of pattern searches introduced by Hooke and Jeeves in [12]. The pattern

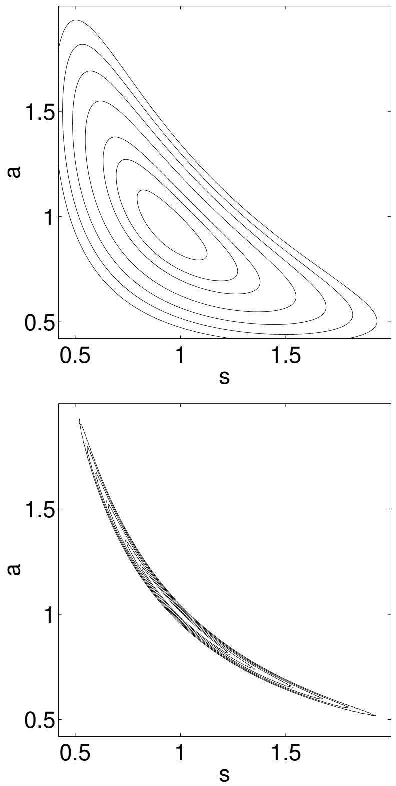

Figure 1: Contour plots of the posterior distribution of the parameters of a linear model $x = as + n$ under different noise levels. In the upper subfigure the noise level is relatively high ($\sigma_n^2 = 10^{-1}$) and the posterior is relatively flat. In the lower subfigure the noise level is lower ($\sigma_n^2 = 10^{-3}$) and the posterior is much narrower.

search consists of two phases. In the first, exploratory phase, the objective function is optimized in each coordinate direction separately as usual. This phase is called the phase of *parameter-wise updates*. The updates performed in the parameter-wise update phase are then combined to form a diagonal direction for the *line search*.

Pattern search methods have mostly been abandoned in standard optimization literature in favour of conjugate gradient and other more advanced methods. Using such methods with the variational approach would, however, require significant changes to the existing algorithms and would not utilise the ability to solve independent subproblems easily. The pattern search approach takes advantage of the existing methodology for performing the parameter-wise updates and requires only minor changes as illustrated by the algorithm optimize_pattern below. It is especially noteworthy that the pattern search approach does not need the derivatives of the cost function.

Assume we are optimizing the function $C : \mathbb{R}^n \to \mathbb{R}$ and let $\mathbf{d}_1, \ldots, \mathbf{d}_n$ be the standard basis of $\mathbb{R}^n$. One iteration of the parameter-wise updates can be implemented as follows.

function optimize_parameter-wise($C, \mathbf{z}_1$):
    $\mathbf{z}_2 \leftarrow \mathbf{z}_1$
    for $i = 1, \ldots, n$:
        $\lambda \leftarrow \text{argmin}_\lambda \; C(\mathbf{z}_2 + \lambda \mathbf{d}_i)$
        $\mathbf{z}_2 \leftarrow \mathbf{z}_2 + \lambda \mathbf{d}_i$
    return $\mathbf{z}_2$

In practice the parameter-wise updates are not performed using a general optimization algorithm. In variational approach the problem is split into smaller subproblems that can be solved more easily and that can be utilized here. The updates can also be done for a larger group of parameters at a time rather than for just one parameter.

The complete pattern search scheme is a straightforward extension to this standard algorithm:

function optimize_pattern($C, \mathbf{z}_1$):
    $\mathbf{z}_2 \leftarrow$ optimize_parameter-wise($C, \mathbf{z}_1$)
    $\Delta\mathbf{z} \leftarrow \mathbf{z}_2 - \mathbf{z}_1$
    $\lambda \leftarrow \text{argmin}_\lambda \; C(\mathbf{z}_1 + \lambda \cdot \Delta\mathbf{z})$
    $\mathbf{z}_3 \leftarrow \mathbf{z}_1 + \lambda \cdot \Delta\mathbf{z}$
    return $\mathbf{z}_3$

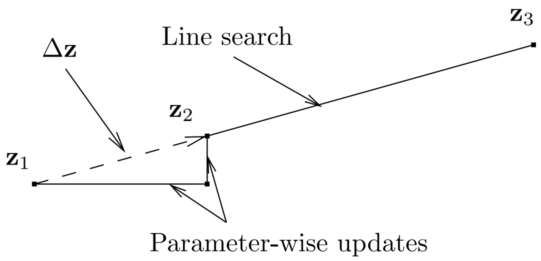

Figure 2: Illustration of the pattern search algorithm.

The algorithm is illustrated in Fig. 2. The figure shows how the direction of the line search is obtained as the difference vector $\Delta\mathbf{z} = \mathbf{z}_2 - \mathbf{z}_1$ of the value of the parameters $\mathbf{z}_2$ after the parameter-wise update round and the corresponding value $\mathbf{z}_1$ before the updates. Parameters such as variances that have a positivity constraint are treated on logarithmic scale for the purposes of evaluating the difference and later when extrapolating for the iterates $\mathbf{z}_1 + \lambda \cdot \Delta\mathbf{z}$. This leads to formulas

$$\Delta\sigma := \log\sigma_2 - \log\sigma_1 = \log\frac{\sigma_2}{\sigma_1} \tag{1}$$

and

$$\sigma_3 = \sigma_1 + \lambda \cdot \Delta\sigma := \exp(\log \sigma_1 + \lambda \cdot \Delta\sigma) \qquad (2)$$

that are used for such parameters.

The figure also shows that although the optimizations in optimize_pattern appear to be similar to those in optimize_parameter-wise, there is a big difference: the line search is made in arbitrary direction instead of standard coordinate directions. In many cases such as ours, the parameter-wise optimizations, i.e. the ones in coordinate directions, can be easily carried out analytically whereas arbitrary line searches are global operations and thus potentially computationally more demanding. The extra work is nevertheless worthwhile as the value of λ can easily be more than 100 and thus a single line search can save at least that many ordinary rounds of optimization.

2.1. Line searches

The line search in optimize_pattern can be implemented by any standard algorithm. As the other parts of the algorithm do not use derivatives of the cost function, it is reasonable to choose a derivative free line search algorithm. Good general candidates for such algorithms are the golden section method and the method of quadratic fit [13, 14].

The golden section method gives decent worst case performance but usually loses to the competitors in more realistic situations. In our examples the cost function appeared roughly quadratic along the line search direction and thus a simple implementation of the method of quadratic fit with golden section method as a fall back for the most difficult cases was used. In practice the best method would probably be some kind of a combination of the two basic methods [13].

2.2. Implementation details

In standard Hooke–Jeeves algorithm there is only one cyclic optimization step between two line searches [14]. This turned out to be suboptimal for our purposes as the benefits of the line searches in increased step length were rather small and the resulting algorithm was even slower than the standard update scheme in some cases.

Letting the iteration stabilize by doing several parameter-wise optimization phases before the next line search seemed to work much better. In the experiments we used ten parameter-wise optimization rounds between two consecutive line search steps. We also tried to interpolate the search direction over several parameter-wise optimization rounds but this did not improve the results.

There are quite a few details in the pattern search method that could be tuned to improve the performance even further. We have not done so as we wish to demonstrate that even as such, the method can provide significant speedups for learning.

3. EXPERIMENTS

We demonstrate the pattern search algorithm with two experiments using variational Bayesian learning. The approximation we use here is also known as ensemble learning [1, 15]. Its key idea is to approximate the exact posterior distribution $p(\boldsymbol{\theta}|\boldsymbol{X})$ by another distribution $q(\boldsymbol{\theta})$ that is computationally easier to handle. The approximating distribution is usually chosen to be a product of several independent distributions, one for each parameter or a set of similar parameters. The optimal approximating distribution is found by minimizing the Kullback-Leibler divergence between the approximate and true posterior.

In the examples considered here, the approximate posterior is a product of independent Gaussian distributions and is thus characterized by the means and variances of these distributions. To ensure validity of the values of the variances, they are treated on logarithmic scale.

3.1. One-dimensional toy example

Let us consider a very simple linear model $x = as + n$ with one-dimensional data x, one-dimensional latent variable s and additive noise n. Though seemingly very trivial, this example shows many of the important problems encountered in more difficult generalizations and it is easy to visualize.

Let us assume that there is a single observation $x = 1$ and we are trying to fit the given linear model $x = as + n$. The variables a and s have Gaussian priors with zero mean and unit variance. Let us use variational Bayesian learning to solve the problem by assuming a and s to be independent. It turns out that the optimal approximate posterior distributions of a and s are Gaussian. Ignoring the variances of the Gaussians, the cost function of posterior means of a and s is very similar to the posterior distributions shown in Fig. 1. For a moderate noise variance of $\sigma_n^2 = 10^{-1}$ in the upper subfigure, it is pretty well-behaved and easy to optimize. When the noise variance decreases, the minimal valley of the cost function becomes narrower and more sharply curved. With noise variance $\sigma_n^2 = 10^{-3}$ in the lower subfigure, the optimization problem is already much more difficult.

The progress of the optimization procedures as a function of time is illustrated in Fig. 3. The cyclic

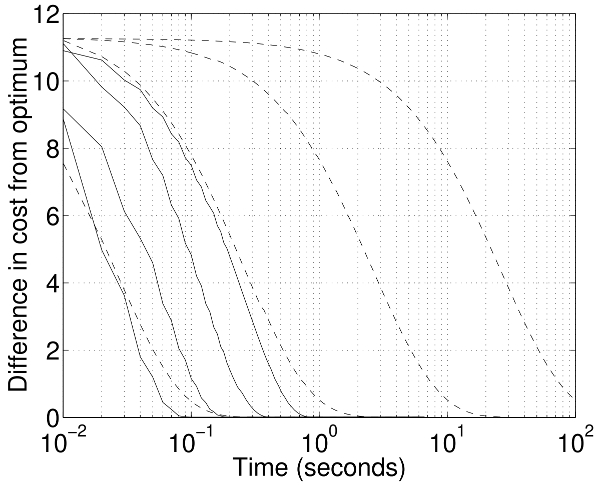

Figure 3: Cost function values attained by different algorithms for the toy example with respect to time under various noise levels. The solid lines represent optimization using pattern searches and dashed lines standard cyclic updates. The noise variance is decreased by factor of 10 in consecutive simulations. The time scale in the figures is logarithmic and thus a constant difference is in fact difference by a constant factor.

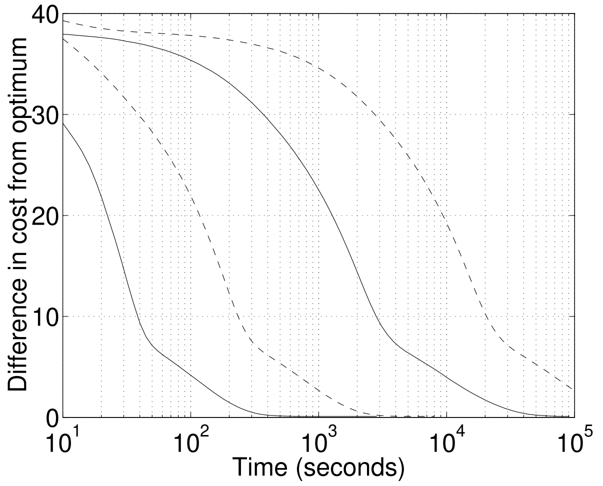

Figure 4: Cost function values attained by different algorithms for the real ICA example with respect to time under two different noise levels. The solid lines represent optimization using pattern searches and dashed lines standard cyclic updates. The noise variance is decreased by factor of 10 in the different simulations. The time scale in the figures is logarithmic and thus a constant difference is in fact difference by a constant factor.

iteration performs very badly in this case, requiring more than a minute to solve the problem with least noise whereas the pattern search algorithm can find the optimum in less than a second. For low noise levels, the length of the steps taken by the iteration is directly proportional to the noise variance. Thus halving the noise level (standard deviation) quadruples the time needed for the algorithm to converge. The pattern search method does much better and slows down considerably less when the noise level decreases.

3.2. Real ICA example

The method was also tested with a more realistic noisy ICA model. The data set used was an eight-dimensional artificial mixture of four source signals. A varying amount of Gaussian noise was added to the data. The number of data points used was 200. The model was a simple linear model with Gaussian mixture priors for the sources. The actual simulations were run using the building block library presented in [8]. Practically all the parameters were estimated directly from the data by the algorithm.

The results of the methods are shown in Fig. 4. The standard update algorithm behaves in a very similar manner as in the simple one-dimensional example. Finding the correct rotation for the sources takes a long time and the time seems to increase inversely proportionally to the variance of the noise added to the data. The pattern search method is again significantly faster, typically reaching the same value of cost function in around 10–20 % of the time needed by the standard method.

4. DISCUSSION

The pattern search algorithm is a straightforward extension of the standard cyclic update scheme but it speeds up the convergence significantly. It is easy to implement and utilizes the ability to solve the independent optimization problems easily. All the operations required scale linearly with respect to the number of parameters in the problem. It could be applied directly to many practical algorithms such as [3, 7, 8, 9].

In optimization literature pattern search methods have, however, mostly been abandoned for more advanced ones like conjugate gradient optimization. The conjugate gradient method is very different from the standard procedure in variational approach as it requires derivatives of the cost function and does not take advantage of the ability to solve the independent subproblems easily. It has nevertheless been used successfully for speeding up the EM algorithm in [16]. It would be interesting to see if the same method would work for other algorithms such as variational Bayesian learning as well, and how it would compare with the pattern search approach. Nevertheless, the conjugate gradient method requires significant algorithmic modifications and thus does not directly compete with the pattern search approach, which attains major improvement in convergence speed with only minor modifications to the standard algorithm.

Acknowledgements

This research has been funded by the European Commission project BLISS, and the Finnish Center of Excellence Programme (2000–2005) under the project New Information Processing Principles. The author wishes to thank Dr. Harri Valpola for useful comments and discussions.

5. REFERENCES

[1] G. Hinton and D. van Camp, "Keeping neural networks simple by minimizing the description length of the weights," in *Proc. of the 6th Ann. ACM Conf. on Computational Learning Theory*, (Santa Cruz, CA, USA), pp. 5–13, 1993.

[2] D. J. C. MacKay, "Ensemble learning for hidden Markov models." Available from `http://wol.ra.phy.cam.ac.uk/mackay/`, 1997.

[3] H. Attias, "Independent factor analysis," *Neural Computation*, vol. 11, no. 4, pp. 803–851, 1999.

[4] M. Jordan, Z. Ghahramani, T. Jaakkola, and L. Saul, "An introduction to variational methods for graphical models," in *Learning in Graphical Models* (M. Jordan, ed.), pp. 105–161, Cambridge, MA, USA: The MIT Press, 1999.

[5] R. M. Neal and G. E. Hinton, "A view of the EM algorithm that justifies incremental, sparse, and other variants," in *Learning in Graphical Models* (M. I. Jordan, ed.), pp. 355–368, Cambridge, MA, USA: The MIT Press, 1999.

[6] Z. Ghahramani and G. Hinton, "Variational learning for switching state-space models," *Neural Computation*, vol. 12, no. 4, pp. 963–996, 2000.

[7] H. Lappalainen and A. Honkela, "Bayesian non-linear independent component analysis by multi-layer perceptrons," in *Advances in Independent Component Analysis* (M. Girolami, ed.), pp. 93–121, Springer-Verlag, 2000.

[8] H. Valpola, T. Raiko, and J. Karhunen, "Building blocks for hierarchical latent variable models," in *Proc. Int. Conf. on Independent Component Analysis and Signal Separation (ICA2001)*, (San Diego, USA), pp. 710–715, 2001.

[9] H. Valpola and J. Karhunen, "An unsupervised ensemble learning method for nonlinear dynamic state-space models," *Neural Computation*, 2002. To appear.

[10] J. C. Bezdek and R. J. Hathaway, "Some notes on alternating optimization," in *Advances in Soft Computing – AFSS 2002* (N. R. Pal and M. Sugeno, eds.), vol. 2275 of *Lecture Notes in Artificial Intelligence*, pp. 288–300, Springer-Verlag, 2002.

[11] A. Hyvärinen, J. Karhunen, and E. Oja, *Independent Component Analysis*. J. Wiley, 2001.

[12] R. Hooke and T. A. Jeeves, "'Direct search' solution of numerical and statistical problems," *J. of the ACM*, vol. 8, no. 2, pp. 212–229, 1961.

[13] R. Fletcher, *Practical Methods of Optimization*. J. Wiley, second ed., 1987.

[14] M. S. Bazaraa, H. D. Sherali, and C. M. Shetty, *Nonlinear Programming: Theory and Algorithms*. J. Wiley, second ed., 1993.

[15] H. Lappalainen and J. Miskin, "Ensemble learning," in *Advances in Independent Component Analysis* (M. Girolami, ed.), pp. 75–92, Springer-Verlag, 2000.

[16] M. Jamshidian and R. I. Jennrich, "Conjugate gradient acceleration of the EM algorithm," *J. of the American Statistical Association*, vol. 88, no. 421, pp. 221–228, 1993.