Combining Conjugate Direction Methods with Stochastic Approximation of Gradients*

Nicol N. Schraudolph **Thore Graepel**
Institute of Computational Sciences
Eidgenössische Technische Hochschule (ETH)
CH-8092 Zürich, Switzerland
http://www.icos.ethz.ch/

Abstract

The method of conjugate directions provides a very effective way to optimize large, deterministic systems by gradient descent. In its standard form, however, it is not amenable to stochastic approximation of the gradient. Here we explore ideas from conjugate gradient in the stochastic (online) setting, using fast Hessian-gradient products to set up low-dimensional Krylov subspaces within individual mini-batches. In our benchmark experiments the resulting online learning algorithms converge orders of magnitude faster than ordinary stochastic gradient descent.

1 INTRODUCTION

1.1 CONJUGATE GRADIENT

For the optimization of large, differentiable systems, algorithms that require the inversion of a curvature matrix (Levenberg, 1944; Marquardt, 1963), or the storage of an iterative approximation of that inverse (quasi-Newton methods such as BFGS — see Press et al., 1992, p. 425ff), are prohibitively expensive. Conjugate gradient methods (Hestenes and Stiefel, 1952), which exactly minimize a d-dimensional unconstrained quadratic problem in d iterations without requiring explicit knowledge of the curvature matrix, have become the method of choice for such problems.

1.2 STOCHASTIC GRADIENT

Empirical loss functions are often minimized using noisy measurements of gradient (and, if applicable, curvature) taken on small, random subsamples ("mini-batches") of data, or even individual data points. This is done for reasons of computational efficiency on large, redundant data sets, and out of necessity when adapting online to a continual stream of noisy, potentially non-stationary data. Unfortunately the fast convergence of conjugate gradient breaks down when the function to be optimized is noisy, since this makes it impossible to maintain the conjugacy of search directions over multiple iterations. The state of the art for such *stochastic* problems is therefore simple gradient descent, coupled with adaptation of local step size and/or momentum parameters.

1.3 EFFICIENT CURVATURE MATRIX-VECTOR PRODUCTS

The most advanced parameter adaptation methods for stochastic gradient descent (Orr, 1995; Schraudolph, 1999, 2002; Graepel and Schraudolph, 2002) rely on fast curvature matrix-vector products that can be obtained efficiently and automatically (Pearlmutter, 1994; Schraudolph, 2002). Their calculation does *not* require explicit storage of the Hessian, which would be $O(d^2)$; the same goes for other measures of curvature, such as the Gauss-Newton approximation of the Hessian, and the Fisher information matrix (Schraudolph, 2002). *Algorithmic differentiation* software[1] provides generic implementations of the building blocks from which these algorithms are constructed. Here we employ these techniques to efficiently compute Hessian-gradient products which we use to implement a stochastic conjugate direction method.

2 STOCHASTIC QUADRATIC OPTIMIZATION

2.1 DETERMINISTIC BOWL

The d-dimensional quadratic bowl provides us with a simplified test setting in which every aspect of the op-

Proceedings of the Ninth International Workshop on Artificial Intelligence and Statistics, to appear Jan. 2003.

[1] See http://www-unix.mcs.anl.gov/autodiff/

timization can be controlled. It is defined by the unconstrained problem of minimizing with respect to d parameters $\boldsymbol{w}$ the function

$$f(\boldsymbol{w}) \;=\; \frac{1}{2}\,(\boldsymbol{w}-\boldsymbol{w}^*)^T \boldsymbol{J}\boldsymbol{J}^T\,(\boldsymbol{w}-\boldsymbol{w}^*)\,, \qquad (1)$$

where the Jacobian $\boldsymbol{J}$ is a $d \times d$ matrix, and $\boldsymbol{w}^*$ the location of the minimum, both of our choosing. By definition the Hessian $\bar{\boldsymbol{H}} = \boldsymbol{J}\boldsymbol{J}^T$ is positive semidefinite and constant with respect to the parameters $\boldsymbol{w}$; these are the two crucial simplifications compared to more realistic, non-linear problems. The gradient here is $\bar{\boldsymbol{g}} = \nabla f(\boldsymbol{w}) = \bar{\boldsymbol{H}}(\boldsymbol{w}-\boldsymbol{w}^*)$.

2.2 STOCHASTIC BOWL

The stochastic optimization problem analogous to the deterministic one above is the minimization (again with respect to $\boldsymbol{w}$) of the function

$$f(\boldsymbol{w},\boldsymbol{X}) = \frac{1}{2\,b}\,(\boldsymbol{w}-\boldsymbol{w}^*)^T \boldsymbol{J}\boldsymbol{X}\boldsymbol{X}^T\boldsymbol{J}^T(\boldsymbol{w}-\boldsymbol{w}^*)\,, \qquad (2)$$

where $\boldsymbol{X} = [\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots \boldsymbol{x}_b]$ is a $d \times b$ matrix collecting a *batch* of b random input vectors to the system, each drawn i.i.d. from a normal distribution: $\boldsymbol{x}_i \sim N(\boldsymbol{0}, \boldsymbol{I})$. This means that $E[\boldsymbol{X}\boldsymbol{X}^T] = b\,\boldsymbol{I}$, so that in expectation this is identical to the deterministic formulation given in (1):

$$E_{\boldsymbol{X}}[f(\boldsymbol{w},\boldsymbol{X})] \;= \\ \frac{1}{2\,b}\,(\boldsymbol{w}-\boldsymbol{w}^*)^T \boldsymbol{J} E[\boldsymbol{X}\boldsymbol{X}^T]\boldsymbol{J}^T(\boldsymbol{w}-\boldsymbol{w}^*) = f(\boldsymbol{w})\,. \qquad (3)$$

The optimization problem is harder here since the objective can only be probed by supplying stochastic inputs to the system, giving rise to the noisy estimates $\boldsymbol{H} = b^{-1}\boldsymbol{J}\boldsymbol{X}\boldsymbol{X}^T\boldsymbol{J}^T$ and $\boldsymbol{g} = \nabla_{\boldsymbol{w}} f(\boldsymbol{w},\boldsymbol{X}) = \boldsymbol{H}(\boldsymbol{w}-\boldsymbol{w}^*)$ of the true Hessian $\bar{\boldsymbol{H}}$ and gradient $\bar{\boldsymbol{g}}$, respectively. The degree of stochasticity is determined by the batch size b; the system becomes deterministic in the limit as $b \to \infty$.

2.3 LINE SEARCH

A common optimization technique is to first determine a search direction, then look for the optimum in that direction. In a quadratic bowl, the step from $\boldsymbol{w}$ to the minimum along direction $\boldsymbol{v}$ is given by

$$\Delta\boldsymbol{w} \;=\; -\,\frac{\boldsymbol{g}^T\boldsymbol{v}}{\boldsymbol{v}^T\boldsymbol{H}\boldsymbol{v}}\,\boldsymbol{v}\,. \qquad (4)$$

$\boldsymbol{H}\boldsymbol{v}$ can be calculated very efficiently (Pearlmutter, 1994), and we can use (4) in stochastic settings as well. Line search in the gradient direction, $\boldsymbol{v} = \boldsymbol{g}$, is called *steepest descent*. When fully stochastic ($b = 1$), steepest descent degenerates into the *normalized LMS* method known in signal processing.

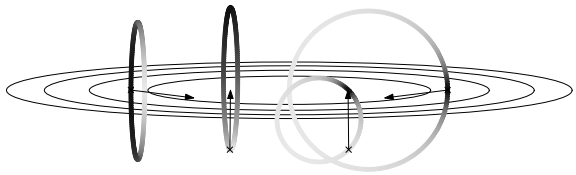

Figure 1: Distribution of stochastic gradient steps from equivalent starting points (crosses) with (circles, right) *vs.* without (ellipses, left, scaled arbitrarily) line search in an ill-conditioned quadratic bowl. Black is high, white low probability density. Compare to deterministic steepest descent (arrows).

2.4 CHOICE OF JACOBIAN

For our experiments we choose $\boldsymbol{J}$ such that the Hessian has a) eigenvalues of widely differing magnitude, and b) eigenvectors of intermediate sparsity. These conditions model the mixture of axis-aligned and oblique "narrow valleys" that is characteristic of multi-layer perceptrons, and a primary cause of the difficulty in optimizing such systems. We achieve them by imposing some sparsity on the notoriously ill-conditioned *Hilbert matrix*, defining

$$(\boldsymbol{J})_{ij} \;\stackrel{\text{def}}{=}\; \begin{cases} \frac{1}{i+j-1} & \text{if } \; i \bmod j \;=\; 0 \\ & \text{or } \; j \bmod i \;=\; 0\,, \\ 0 & \text{otherwise}\,. \end{cases} \qquad (5)$$

We call the optimization problem resulting from setting $\boldsymbol{J}$ to this matrix the *modified Hilbert bowl*. In the experiments reported here we used the modified Hilbert bowl of dimension $d = 5$, which has a condition number of $4.9 \cdot 10^3$.

2.5 STOCHASTIC ILL-CONDITIONING

Such ill-conditioned systems are particularly challenging for stochastic gradient descent. While directions associated with large eigenvalues are rapidly optimized, progress along the floor of the valley spanned by the small eigenvalues is extremely slow. Line search can ameliorate this problem by amplifying small gradients, but for this to happen the search direction must lie along the valley floor in the first place. In a stochastic setting, gradients in that direction are not just small but extremely *unlikely*: in contrast to deterministic gradients, stochastic gradients contain large components in directions associated with large eigenvalues even for points right at the bottom of the valley. Figure 1 illustrates (for $b = 1$) the consequence: although a line search can stretch the narrow ellipses of possible stochastic gradient steps into circles through the minimum, it cannot shift any probability mass in that direction.

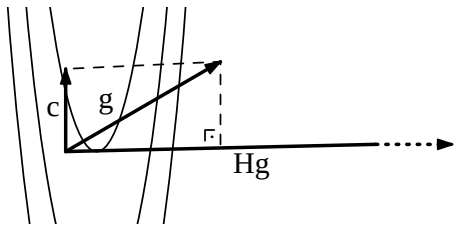

Figure 2: Construction of the conjugate direction $\boldsymbol{c}$ by subtracting from gradient $\boldsymbol{g}$ its projection onto $\boldsymbol{Hg}$.

3 STOCHASTIC CONJUGATE DIRECTIONS

Looking for ways to improve the convergence of stochastic gradient methods in narrow valleys, we note that relevant directions associated with large eigenvalues can be identified by multiplying the (stochastic estimates of) Hessian $\boldsymbol{H}$ and gradient $\boldsymbol{g}$ of the system. Subtracting the *projection* of $\boldsymbol{g}$ onto $\boldsymbol{Hg}$ from $\boldsymbol{g}$ (Figure 2) then yields a *conjugate* descent direction $\boldsymbol{c}$ that emphasizes directions associated with small eigenvalues, by virtue of being orthogonal to $\boldsymbol{Hg}$:

$$\boldsymbol{c} \;=\; \boldsymbol{g} \;-\; \frac{\boldsymbol{g}^T\boldsymbol{Hg}}{\boldsymbol{g}^T\boldsymbol{HHg}}\,\boldsymbol{Hg} \tag{6}$$

Figure 3 shows that stochastic descent in direction of $\boldsymbol{c}$ (dotted) indeed sports much better late convergence than steepest descent (dashed). Since directions with large eigenvalues are subtracted out, however, it takes far longer to reach the valley floor in the first place.

3.1 TWO-DIMENSIONAL METHOD

We can combine the respective strengths of gradient and conjugate direction by performing, at each stochastic iteration, a two-dimensional minimization in the plane spanned by $\boldsymbol{g}$ and $\boldsymbol{Hg}$. That is, we seek the α_1, α_2 that produce the optimal step

$$\Delta\boldsymbol{w} \;=\; \alpha_1\boldsymbol{g} \;+\; \alpha_2\boldsymbol{Hg}\,. \tag{7}$$

Using $\boldsymbol{g} \stackrel{\text{def}}{=} \boldsymbol{H}(\boldsymbol{w}-\boldsymbol{w}^*)$ gives $\Delta\boldsymbol{g} = \alpha_1\boldsymbol{Hg} + \alpha_2\boldsymbol{HHg}$. We now express the optimality condition as a system of linear equations in the quadratic forms $q_i \stackrel{\text{def}}{=} \boldsymbol{g}^T\boldsymbol{H}^i\boldsymbol{g}$:

$$\boldsymbol{g}^T(\boldsymbol{g}+\Delta\boldsymbol{g}) \;=\; q_0 \;+\; \alpha_1\,q_1 \;+\; \alpha_2\,q_2 \;\stackrel{!}{=}\; 0 \tag{8}$$

$$\boldsymbol{g}^T\boldsymbol{H}\,(\boldsymbol{g}+\Delta\boldsymbol{g}) \;=\; q_1 \;+\; \alpha_1\,q_2 \;+\; \alpha_2\,q_3 \;\stackrel{!}{=}\; 0 \tag{9}$$

Solving this yields

$$\alpha_1 \;=\; \frac{q_0q_3 - q_1q_2}{q_2^2 - q_1q_3}\,, \quad \alpha_2 \;=\; \frac{q_1^2 - q_0q_2}{q_2^2 - q_1q_3} \tag{10}$$

Figure 3 shows that this technique (dot-dashed) indeed combines the advantages of the gradient and conjugate directions.

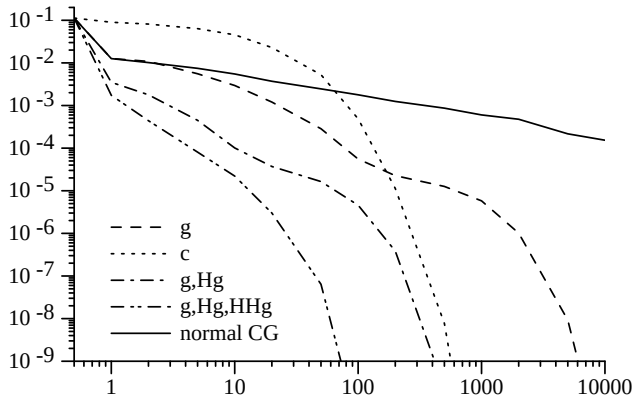

Figure 3: Log-log plot of average loss (over 100 runs) *vs.* number of stochastic iterations ($b = 3$) in modified Hilbert bowl when minimizing in: direction of $\boldsymbol{g}$ (dashed), direction of $\boldsymbol{c}$ (dotted), plane spanned by $\boldsymbol{g}$ and $\boldsymbol{Hg}$ (dot-dashed), and subspace spanned by $\boldsymbol{g}$, $\boldsymbol{Hg}$, and $\boldsymbol{HHg}$ (dot-dot-dashed). Compare to normal conjugate gradient across mini-batches (solid line).

3.2 STOCHASTIC KRYLOV SUBSPACE

This approach can be extended to performing minimizations in the m-dimensional, stochastic *Krylov subspace* $K_m \stackrel{\text{def}}{=} [\boldsymbol{g}, \boldsymbol{Hg}, \ldots \boldsymbol{H}^{m-1}\boldsymbol{g}]$ with $m \le \min(d, b)$. The expansion of $\Delta\boldsymbol{g}$ in K_m is given by

$$\Delta\boldsymbol{g} \;=\; \sum_{i=1}^{m} \alpha_i\,\boldsymbol{H}^i\boldsymbol{g}\,; \tag{11}$$

for optimality we require

$$\boldsymbol{q} \;+\; \boldsymbol{Q}\,\boldsymbol{\alpha} \;\stackrel{!}{=}\; 0 \quad\Rightarrow\quad \boldsymbol{\alpha} \;=\; -\,\boldsymbol{Q}^{-1}\boldsymbol{q} \tag{12}$$

with

$$\boldsymbol{q} \;\stackrel{\text{def}}{=}\; \begin{bmatrix} q_0 \\ q_1 \\ \vdots \\ q_{m-1} \end{bmatrix}, \quad \boldsymbol{\alpha} \;\stackrel{\text{def}}{=}\; \begin{bmatrix} \alpha_1 \\ \alpha_2 \\ \vdots \\ \alpha_m \end{bmatrix},$$

$$\boldsymbol{Q} \;\stackrel{\text{def}}{=}\; \begin{bmatrix} q_1 & q_2 & \cdots & q_m \\ q_2 & q_3 & \cdots & q_{m+1} \\ \vdots & \vdots & \ddots & \vdots \\ q_m & q_{m+1} & \cdots & q_{2m-1} \end{bmatrix} \tag{13}$$

$\boldsymbol{Q}$ and $\boldsymbol{\alpha}$ can be flipped to bring (12) into the form of a standard Toeplitz system, which can be solved in as little as $O\big(m\,\log^2 m\big)$ operations (Press et al., 1992, p. 92ff). One can either do this numerically at each iteration, or simply use a precomputed analytic solution. The analytic solution of the first-order system (*i.e.*, steepest descent) is $\alpha_1 = -q_0/q_1$, as evident from (4); for $m = 2$ it is given in (10); for the third-order

system it is

$$\begin{aligned}
\alpha_1 u &= q_0 q_4^2 + q_1 q_2 q_5 + q_2 q_3^2 - q_0 q_3 q_5 - q_1 q_3 q_4 - q_4 q_2^2, \\
\alpha_2 u &= q_1 q_2 q_4 + q_0 q_2 q_5 + q_1 q_3^2 - q_5 q_1^2 - q_3 q_2^2 - q_0 q_3 q_4, \\
\alpha_3 u &= q_0 q_3^2 + q_1^2 q_4 + q_2^3 - q_0 q_2 q_4 - 2 q_1 q_2 q_3, \text{ with} \\
u &= q_1 q_3 q_5 + 2 q_2 q_3 q_4 - q_1 q_4^2 - q_2^2 q_5 - q_3^3. \qquad (14)
\end{aligned}$$

The quadratic forms q_0 through q_{2m-1} required by either solution strategy can be calculated efficiently as inner products of the m fast Hessian-vector products $\boldsymbol{H}^i \boldsymbol{g}\,, 0 \leq i \leq m$. Figure 3 (dot-dot-dashed) illustrates the rapid convergence of this approach for $m = 3$.

3.3 RELATION TO CONJUGATE GRADIENT METHODS

It has not escaped our notice that on quadratic optimization problems, instead of solving this linear system of equations explicitly, we can equivalently perform m steps of ordinary conjugate gradient within each mini-batch to find the optimum in the Krylov subspace K_m. Instead of a single m-dimensional optimization, we then have m successive line searches according to (4). The initial search direction is set to the gradient, $\boldsymbol{v}_0 := \boldsymbol{g}$; subsequent ones are calculated via the formula

$$\boldsymbol{v}_{t+1} \;=\; \frac{\boldsymbol{g}_{t+1}^T \boldsymbol{H} \boldsymbol{v}_t}{\boldsymbol{v}_t^T \boldsymbol{H} \boldsymbol{v}_t}\, \boldsymbol{v}_t \;-\; \boldsymbol{g}_{t+1} \qquad (15)$$

or one of its well-known variants (such as Fletcher-Reeves or Polak-Ribiere — see Press et al., 1992, p. 420ff). The crucial difference to standard conjugate gradient techniques is that here we propose to perform just a few steps of conjugate gradient *within* each small, stochastic mini-batch. A reset to the gradient direction when moving on to another mini-batch is not only recommended but indeed mandatory for our approach to work, since otherwise the stochasticity collapses the Krylov subspace. To illustrate, we show in Figure 3 (solid line) the inadequate performance of standard conjugate gradient when misapplied in this fashion.

3.4 NON-REALIZABLE PROBLEMS

Our stochastic quadratic bowl (2) models *realizable* problems, *i.e.*, those where the loss at the optimum reaches zero for all inputs:

$$(\forall \boldsymbol{X})\; f(\boldsymbol{w}^*, \boldsymbol{X}) \;=\; 0 \qquad (16)$$

Of greater practical relevance are non-realizable problems, in which the optimum carries a non-zero loss reflecting the best compromise between conflicting demands placed on the model by the data. We model this by incorporating a multivariate Gaussian random vector $\boldsymbol{r}$ with zero mean and variance $E[\boldsymbol{r}\boldsymbol{r}^T] = \sigma^2 \boldsymbol{I}$ into our objective, which now becomes

$$\begin{aligned}
f_\sigma(\boldsymbol{w}, \boldsymbol{X}) &= \frac{1}{2\,b}\, \boldsymbol{e}^T \boldsymbol{e}\,, \text{ where} \qquad (17) \\
\boldsymbol{e} &\stackrel{\text{def}}{=} \boldsymbol{X}^T \boldsymbol{J}^T (\boldsymbol{w} - \boldsymbol{w}^*) \;+\; \boldsymbol{r}\,.
\end{aligned}$$

This makes the expected loss at the optimum $\boldsymbol{w}^*$ be

$$E[f_\sigma(\boldsymbol{w}^*, \boldsymbol{X})] \;=\; \frac{1}{2\,b}\, E[\boldsymbol{r}^T \boldsymbol{r}] \;=\; \tfrac{1}{2}\,\sigma^2. \qquad (18)$$

Moreover, the presence of $\boldsymbol{r}$ makes it impossible to determine $\boldsymbol{w}^*$ precisely from a finite data sample; the smaller the sample size b, the greater (for a given σ) the uncertainty in $\boldsymbol{w}^*$. Conventional stochastic gradient methods address this issue by multiplying the gradient step with a rate factor that is gradually annealed towards zero, thus effectively averaging the gradient over progressively larger stretches of data.

Here, however, we invest significantly greater computational effort (proportional to the order m of our Krylov subspace) into learning from each given batch of data; taking only partial — and, in the limit, infinitesimal — steps towards the inferred goal is therefore not attractive. Instead, we propose to adaptively increase the batch size b over time. Specifically, whenever we fail to reduce the loss (compared to the previous batch of data), we know that this is due to the uncertainty in $\boldsymbol{w}^*$, and consequently double the batch size b to obtain a better estimate.

Figure 4 shows the performance of the resulting algorithm (solid) on the non-realizable modified Hilbert

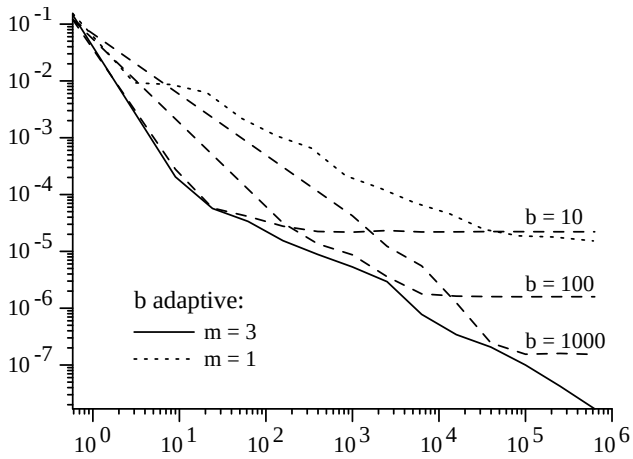

Figure 4: Log-log plot of average excess loss (over 10 runs) on the non-realizable modified Hilbert bowl against number of cycles for our stochastic Krylov subspace method ($m = 3$) with adaptive (solid) *vs.* various fixed batch sizes (dashed). Also shown: steepest descent ($m = 1$) with adaptive batch size (dotted).

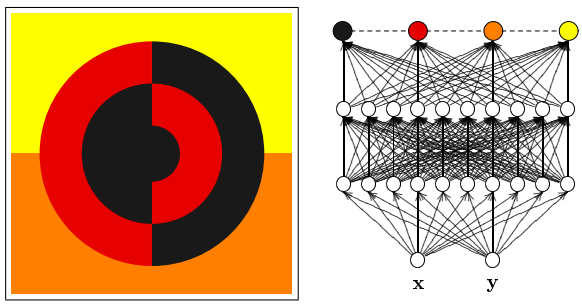

Figure 5: The four regions benchmark (left), and the neural network trained on it (right).

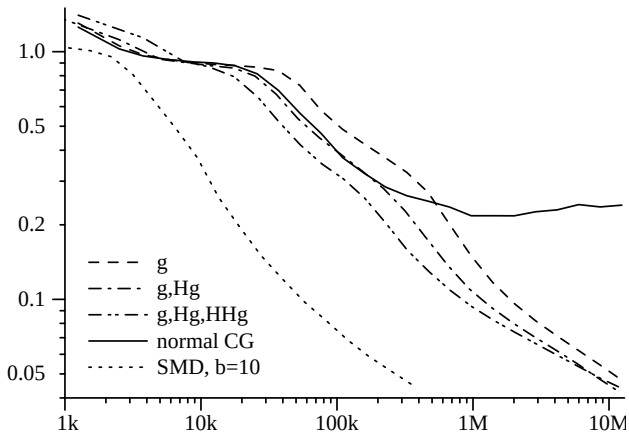

Figure 6: Log-log plot of average loss (over 10 runs) on the four regions problem against number of cycles for our stochastic Krylov subspace method up to $m = 3$. Compare to normal conjugate gradient (solid) and stochastic meta-descent with $b = 10$ (dotted).

bowl with $\sigma = 10^{-2}$, $d = 5$, $m = 3$, and initial batch size $b_0 = 10$. For fair comparison to various fixed batch sizes (dashed) as well as steepest descent (dotted), we plot excess loss — *i.e.*, that above $f(\boldsymbol{w}^*)$ — against the number of *cycles*, obtained by multiplying the number of data points seen with the Krylov subspace order m.

For this stationary optimization problem, our simple batch size adaptation heuristic is successful in obtaining the rapid initial convergence characteristic of small batch sizes while eliminating the noise floor that limits asymptotic performance for any fixed batch size. We also see that our stochastic Krylov subspace approach continues to perform well for this non-realizable case in that it greatly outperforms steepest descent; its convergence could be further hastened by increasing the subspace order, up to $m = \min(b, d)$. Finally, we note in passing that as expected for our stochastic setting, the naive implementation of conjugate gradient — that is, a single iteration per batch, with no reset between batches — performs very poorly here.

3.5 NON-LINEAR PROBLEMS

An extension of our techniques to non-linear optimization problems, such as online learning in multi-layer perceptrons, raises additional questions. In this case, conjugate gradient methods are *not* equivalent to the explicit solution of (12), raising the question which is preferable in the stochastic gradient setting. Both approaches must be protected against near-zero or negative eigenvalues of $\boldsymbol{H}$ (Møller, 1993; Schraudolph, 2002). Here we report on our first experiments designed to begin exploring these issues.

3.5.1 Four Regions Benchmark

We are using the "four regions" benchmark (Singhal and Wu, 1989), on which we have extensive experience with accelerated stochastic gradient methods (Schraudolph, 1999, 2002; Graepel and Schraudolph, 2002). A fully connected feedforward multi-layer perceptron with two hidden layers of 10 units each (Figure 5, right) is to classify two continuous inputs in the range [-1,1] into four disjoint, non-convex regions (Figure 5, left). We use the standard softmax output nonlinearity with cross-entropy loss function (Bishop, 1995, chapter 6), and hyperbolic tangent hidden units.

For each run the 184 weights (including bias weights for all units) are initialized to uniformly random values in the range [-0.3,0.3]. Training patterns are generated online by drawing independent, uniformly random input samples. To obtain an unbiased estimate of generalization ability, we record the network's loss on each batch of samples *before* using it for optimization.

Since our non-linear optimization setup does not yet permit batch size adaptation, we fix the batch size at $b = 500$. To avoid near-zero or negative eigenvalues of $\boldsymbol{H}$ we instead use $\boldsymbol{G} + \lambda \boldsymbol{I}$ throughout, where $\boldsymbol{G}$ is an extended Gauss-Newton approximation of the Hessian (Schraudolph, 2002). Since it is not clear yet whether (and how) methods that vary the trust region parameter λ (*e.g.*, Møller, 1993) can be adapted to our stochastic setting, we simply use the smallest fixed value found to provide reasonably stable convergence, namely $\lambda = 10$.

3.5.2 Results

Figure 6 shows that even a batch size of $b = 500$, the remaining stochasticity in the problem causes conventional conjugate gradient (solid line) to peter out at a loss of about 0.25. By contrast, our stochastic Krylov subspace method — implemented via explicit solution of the Toeplitz system (12) — continues to reduce the loss, converging at a rate that increases with order

m. Due to the large batch size, however, it does not compare well to stochastic meta-descent (Schraudolph, 1999), a local learning rate adaptation method that makes do with far smaller mini-batches (dotted line; parameters: $b = 10, \lambda = 0.998, \mu = 0.05, \boldsymbol{p}_0 = 0.1$). A further increase of m may help narrow this gap.

We also found that performing 2-3 steps of conjugate gradient within each batch, using (4) as a simple line search, resulted in performance even worse than conventional conjugate gradient (Figure 6, solid line). A more accurate, iterative line search is bound to improve this, albeit at significant computational expense; directly solving the Toeplitz system — which does not involve costly line searches — may well be preferable.

4 SUMMARY AND OUTLOOK

We considered the problem of stochastic ill-conditioning of optimization problems that leads to inefficiency in standard stochastic gradient methods. By geometric arguments we arrived at conjugate search directions that can be found efficiently by fast Hessian-vector products. The resulting algorithm can be interpreted as a stochastic conjugate gradient technique and as such introduces Krylov subspace methods into stochastic optimization. Numerical results show that our approach outperforms standard gradient descent for unconstrained quadratic optimization problems — realizable and non-realizable — by orders of magnitude in the stochastic setting where standard conjugate gradient fails. First experiments with the four regions benchmark indicate that our approach improves upon standard conjugate gradient methods for non-linear problems as well.

We are now working to advance our implementation of non-linear optimization by incorporating adaptation of batch size and trust region parameters, as well as a Toeplitz solver for orders $m > 3$. We are also interested in developing better methods for controlling batch size and subspace order than the simple heuristic employed in Section 3.4.

Acknowledgments

We want to thank Gert Lanckriet for our discussion at the 2002 Learning workshop, the anonymous referees for helpful comments, and the ETH Zürich for financial support. An earlier account of the linear, realizable case can be found in (Schraudolph and Graepel, 2002).

References

C. M. Bishop. *Neural Networks for Pattern Recognition*. Clarendon Press, Oxford, 1995.

J. R. Dorronsoro, editor. *Proc. Intl. Conf. Artificial Neural Networks*, volume 2415 of *Lecture Notes in Computer Science*, Madrid, Spain, 2002. Springer Verlag, Berlin.

T. Graepel and N. N. Schraudolph. Stable adaptive momentum for rapid online learning in nonlinear systems. In Dorronsoro (2002), pages 450–455. `http://www.inf.ethz.ch/~schraudo/pubs/sam.ps.gz`.

M. R. Hestenes and E. Stiefel. Methods of conjugate gradients for solving linear systems. *Journal of Research of the National Bureau of Standards*, 49:409–436, 1952.

K. Levenberg. A method for the solution of certain non-linear problems in least squares. *Quarterly Journal of Applied Mathematics*, II(2):164–168, 1944.

D. Marquardt. An algorithm for least-squares estimation of non-linear parameters. *Journal of the Society of Industrial and Applied Mathematics*, 11(2): 431–441, 1963.

M. F. Møller. A scaled conjugate gradient algorithm for fast supervised learning. *Neural Networks*, 6(4): 525–533, 1993.

G. B. Orr. *Dynamics and Algorithms for Stochastic Learning*. PhD thesis, Department of Computer Science and Engineering, Oregon Graduate Institute, Beaverton, OR 97006, 1995. `ftp://neural.cse.ogi.edu/pub/neural/papers/orrPhDch1-5.ps.Z`, `orrPhDch6-9.ps.Z`.

B. A. Pearlmutter. Fast exact multiplication by the Hessian. *Neural Computation*, 6(1):147–160, 1994.

W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery. *Numerical Recipes in C: The Art of Scientific Computing*. Cambridge University Press, second edition, 1992.

N. N. Schraudolph. Local gain adaptation in stochastic gradient descent. In *Proc. Intl. Conf. Artificial Neural Networks*, pages 569–574, Edinburgh, Scotland, 1999. IEE, London. `http://www.inf.ethz.ch/~schraudo/pubs/smd.ps.gz`.

N. N. Schraudolph. Fast curvature matrix-vector products for second-order gradient descent. *Neural Computation*, 14(7):1723–1738, 2002. `http://www.inf.ethz.ch/~schraudo/pubs/mvp.ps.gz`.

N. N. Schraudolph and T. Graepel. Conjugate directions for stochastic gradient descent. In Dorronsoro (2002), pages 1351–1356. `http://www.inf.ethz.ch/~schraudo/pubs/hg.ps.gz`.

S. Singhal and L. Wu. Training multilayer perceptrons with the extended Kalman filter. In D. S. Touretzky, editor, *Advances in Neural Information Processing Systems. Proceedings of the 1988 Conference*, pages 133–140, San Mateo, CA, 1989. Morgan Kaufmann.