Conjugate Directions for Stochastic Gradient Descent*

Nicol N. Schraudolph   Thore Graepel

Institute of Computational Science
ETH Zürich, Switzerland
{schraudo,graepel}@inf.ethz.ch

Abstract. The method of conjugate gradients 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

Conjugate gradient. For the optimization of large, differentiable systems, algorithms that require the inversion of a curvature matrix (*e.g.*, Levenberg-Marquardt [1, 2]), or the storage of an iterative approximation of that inverse (quasi-Newton methods such as BFGS [9, p. 425ff]), are prohibitively expensive. Conjugate gradient methods [3], 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.

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.

Curvature matrix-vector products. The most advanced parameter adaptation methods [4–7] for stochastic gradient descent rely on fast curvature matrix-vector products that can be obtained efficiently and automatically [7, 8]. Their calculation does *not* require explicit storage of the Hessian, which would be

* Proc. Intl. Conf. Artificial Neural Networks, LNCS, Springer Verlag, Berlin 2002

$O(d^2)$; the same goes for other measures of curvature, such as the Gauss-Newton approximation of the Hessian, and the Fisher information matrix [7]. *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

Deterministic bowl. The d-dimensional quadratic bowl provides us with a simplified test setting in which every aspect of the optimization can be controlled. It is defined by the unconstrained problem of minimizing with respect to d parameters $\boldsymbol{w}$ the function

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

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, nonlinear problems. The gradient here is $\bar{\boldsymbol{g}} = \nabla f(\boldsymbol{w}) = \bar{\boldsymbol{H}}(\boldsymbol{w}-\boldsymbol{w}^*)$.

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

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

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:

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

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 $b \to \infty$.

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

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

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

$\boldsymbol{Hv}$ can be calculated very efficiently [8], 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.

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

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

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 $4.9 \cdot 10^3$.

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. Fig. 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.

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 $\boldsymbol{g}$ (Fig. 2, left) then yields a *conjugate* descent direction $\boldsymbol{c}$ that emphasizes directions associated with small eigenvalues, by virtue of being orthogonal to $\boldsymbol{Hg}$:

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

Fig. 2 (right) shows that stochastic descent in direction of $\boldsymbol{c}$ (dashed) indeed sports much better late convergence than steepest descent (dotted). Since directions with large eigenvalues are subtracted out, however, it takes far longer to reach the valley floor in the first place.

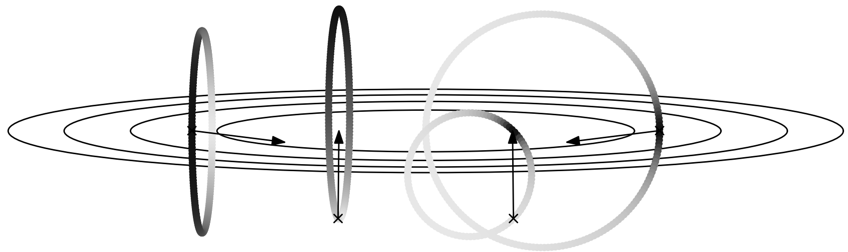

Fig. 1. Distribution of stochastic gradient steps from equivalent points with (circles, right) *vs.* without (ellipses, left) line search in ill-conditioned quadratic bowl. Black is high, white low probability density. Compare to deterministic steepest descent (arrows).

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

$$\Delta \boldsymbol{w} \;=\; \alpha_1 \boldsymbol{g} \;+\; \alpha_2 \boldsymbol{H}\boldsymbol{g}\,. \tag{6}$$

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 can now express the optimality condition as a system of linear equations in the quadratic forms $q_i \stackrel{\text{def}}{=} \boldsymbol{g}^T \boldsymbol{H}^i \boldsymbol{g}$:

$$\boldsymbol{g}^T(\boldsymbol{g} + \Delta\boldsymbol{g}) \;=\; q_0 \,+\, \alpha_1\, q_1 \,+\, \alpha_2\, q_2 \;\stackrel{!}{=}\; 0 \tag{7}$$

$$\boldsymbol{g}^T \boldsymbol{H}\,(\boldsymbol{g} + \Delta\boldsymbol{g}) \;=\; q_1 \,+\, \alpha_1\, q_2 \,+\, \alpha_2\, q_3 \;\stackrel{!}{=}\; 0 \tag{8}$$

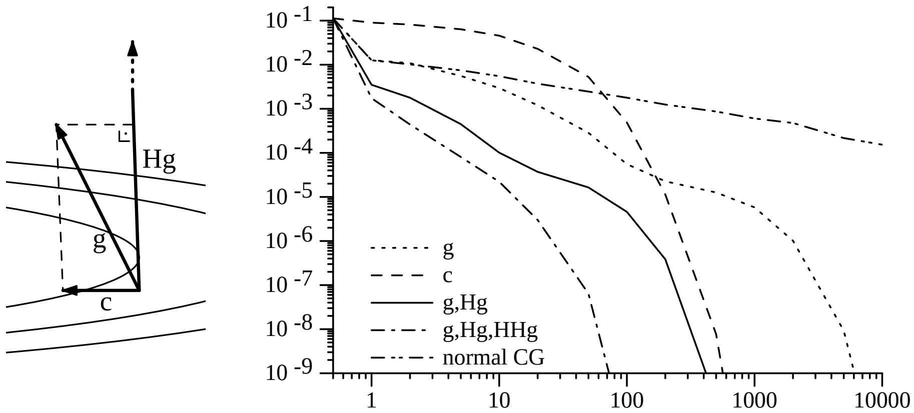

Fig. 2. Left: construction of conjugate direction $\boldsymbol{c}$ via projection of gradient $\boldsymbol{g}$ onto $\boldsymbol{Hg}$. Right: log-log plot of average loss (over 100 runs) *vs.* number of stochastic iterations ($b = 3$) in modified Hilbert bowl when minimizing in: direction of $\boldsymbol{g}$ (dotted), direction of $\boldsymbol{c}$ (dashed), plane spanned by $\boldsymbol{g}$ and $\boldsymbol{Hg}$ (solid), and subspace spanned by $\boldsymbol{g}$, $\boldsymbol{Hg}$, and $\boldsymbol{HHg}$ (dash-dotted). Compare to normal conjugate gradient (dot-dash-dotted).

Solving this yields

$$\alpha_1 \;=\; \frac{q_0 q_3 - q_1 q_2}{q_2^2 - q_1 q_3}\,, \quad \alpha_2 \;=\; \frac{q_1^2 - q_0 q_2}{q_2^2 - q_1 q_3} \tag{9}$$

Fig. 2 (right) shows that this approach (solid line) indeed combines the advantages of the gradient and conjugate directions.

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

$$\Delta\boldsymbol{g} \;=\; \sum_{i=1}^{m} \alpha_i\, \boldsymbol{H}^i \boldsymbol{g}\,; \tag{10}$$

for optimality we require

$$\boldsymbol{q} + \boldsymbol{Q}\,\boldsymbol{\alpha} \stackrel{!}{=} 0 \quad \Rightarrow \quad \boldsymbol{\alpha} \;=\; -\boldsymbol{Q}^{-1}\boldsymbol{q} \tag{11}$$

with

$$\boldsymbol{q} \stackrel{\text{def}}{=} \begin{bmatrix} q_0 \\ q_1 \\ \vdots \\ q_{m-1} \end{bmatrix}, \quad \boldsymbol{\alpha} \stackrel{\text{def}}{=} \begin{bmatrix} \alpha_1 \\ \alpha_2 \\ \vdots \\ \alpha_m \end{bmatrix}, \quad \boldsymbol{Q} \stackrel{\text{def}}{=} \begin{bmatrix} q_1 & q_2 & \cdots & q_m \\ q_2 & q_3 & \cdots & q_{m+1} \\ \vdots & \vdots & \ddots & \vdots \\ q_m & q_{m+1} & \cdots & q_{2m-1} \end{bmatrix} \tag{12}$$

$\boldsymbol{Q}$ and $\boldsymbol{\alpha}$ can be flipped to bring (11) into the form of a standard Toeplitz system, which can be solved in as little as $O\big(m \log^2 m\big)$ operations [9, p. 92ff]. The quadratic forms q_0 through q_{2m} can be calculated efficiently as inner products of the m fast Hessian-vector products $\boldsymbol{H}^i\boldsymbol{g}\,, 0 \leq i \leq m$. Fig. 2 (right) illustrates the rapid convergence of this approach for $m = 3$ (dash-dotted).

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

$$\boldsymbol{v}_{t+1} \;=\; \frac{\boldsymbol{g}_{t+1}^T \boldsymbol{H}\boldsymbol{v}_t}{\boldsymbol{v}_t^T \boldsymbol{H}\boldsymbol{v}_t}\,\boldsymbol{v}_t \;-\; \boldsymbol{g}_{t+1} \tag{13}$$

or one of its well-known variants (Fletcher-Reeves, Polak-Ribiere [9, 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 Fig. 2 (dot-dash-dotted) the inadequate performance of standard conjugate gradient when misapplied in this fashion.

4 Summary and Outlook

We considered the problem of stochastic ill-conditioning of optimization problems that lead 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 by orders of magnitude in a noisy scenario where standard conjugate gradient fails.

At present we only consider sampling noise due to small batch size. Future work will address the question of noise in the target vector $\boldsymbol{w}^*$, corresponding to unrealizable approximation problems, which may require the incorporation of some form of step size annealing or adaptation scheme. We are also investigating the extension of our techniques to nonlinear optimization problems, such as online learning in multi-layer perceptrons. In this case, conjugate gradient methods are *not* equivalent to the explicit solution of (11), and it is an open question which approach is preferable in the stochastic gradient setting.

References

[1] 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.

[2] 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.

[3] 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.

[4] 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`.

[5] T. Graepel and N. N. Schraudolph. Stable adaptive momentum for rapid online learning in nonlinear systems. In *Proceedings of the International Conference on Artificial Neural Networks* (to appear), Lecture Notes in Computer Science. Springer Verlag, Berlin, 2002.

[6] N. N. Schraudolph. Local gain adaptation in stochastic gradient descent. In *Proceedings of the 9th International Conference on Artificial Neural Networks*, pages 569–574, Edinburgh, Scotland, 1999. IEE, London. `http://www.inf.ethz.ch/~schraudo/pubs/smd.ps.gz`.

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

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

[9] 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.