Nonlinear Conjugate Gradients

Aleksandr Artemenkov artemenkov.aa@phystech.edu
Optimization Class Project. MIPT

Nonlinear CG Intro

Conjugate Gradients approach can be formulated as follows:

1. Direction generation rule

$$\mathbf{d_0} = -\nabla f(\mathbf{x_0})$$
$$\mathbf{d_k} = -\nabla f(\mathbf{x_k}) + \beta_k \mathbf{d_{k-1}}$$

2. Line search for an appropriate point:

(a) Exact solution (usually too expensive)

$$\mathbf{d_k}^T \nabla f(\mathbf{x_k} + \alpha_k \mathbf{d_k}) = 0$$

(b) Strong Wolfe conditions

$$f(\mathbf{x_k} + \alpha_k \mathbf{d_k}) \leq f(\mathbf{x_k}) + c_1 \alpha_k \mathbf{d_k}^T \nabla f(\mathbf{x_k})$$
$$|\mathbf{d_k}^T \nabla f(\mathbf{x_k} + \alpha_k \mathbf{d_k})| \leq c_2 |\mathbf{d_k}^T \nabla f(\mathbf{x_k})|$$

(c) Weak Wolfe conditions

$$f(\mathbf{x_k} + \alpha_k \mathbf{d_k}) \leq f(\mathbf{x_k}) + c_1 \alpha_k \mathbf{d_k}^T \nabla f(\mathbf{x_k})$$
$$\mathbf{d_k}^T \nabla f(\mathbf{x_k} + \alpha_k \mathbf{d_k}) \geq c_2 \mathbf{d_k}^T \nabla f(\mathbf{x_k})$$

There many possibilities for β_k selection and each of them is equivalent to **linear CG** in case of quadratic loss function with exact line search on each step:

1. Fletcher-Reeves [1]

$$\beta_k^{FR} = \frac{||\mathbf{g_k}||}{||\mathbf{g_{k-1}}||}$$

2. Polak-Ribiere-Polyak (used in *SciPy* [2])

$$\beta_k^{PRP} = \frac{\mathbf{g_k}^T \mathbf{y_k}}{||\mathbf{g_{k-1}}||}, \quad \mathbf{y_k} = \mathbf{g_k} - \mathbf{g_{k-1}}$$

3. Hestenes-Stiefel

$$\beta_k^{HS} = \frac{\mathbf{g_k}^T \mathbf{y_k}}{\mathbf{d_{k-1}}^T \mathbf{y_k}}, \quad \mathbf{y_k} = \mathbf{g_k} - \mathbf{g_{k-1}}$$

4. Dai-Yuan [3]

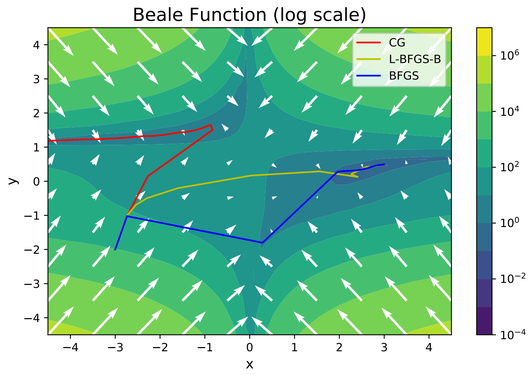
$$\beta_k^{DY} = \frac{||\mathbf{g_k}||}{\mathbf{d_{k-1}}^T \mathbf{y_k}}, \quad \mathbf{y_k} = \mathbf{g_k} - \mathbf{g_{k-1}}$$

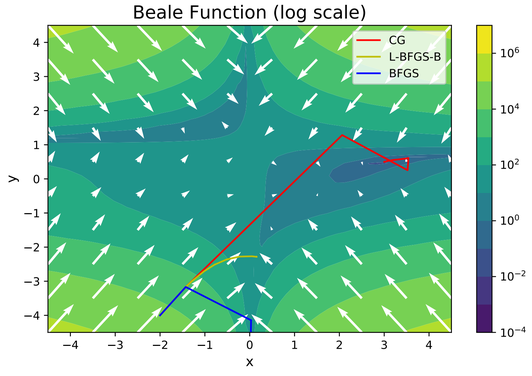
5. Hager-Zhang [4]

$$\beta_k^{HZ} = \frac{1}{\mathbf{d_{k-1}}^T \mathbf{y_k}} \left(\mathbf{y_k} - 2\mathbf{d_{k-1}} \frac{||\mathbf{y_k}||}{\mathbf{d_{k-1}}^T \mathbf{y_k}} \right)^T \mathbf{g_k}$$

Competing BFGS and L-BFGS

The **BFGS** and **L-BFGS** methods are widely used in unconstrained optimization. Moreover, they are often thought to be better than **CG** in average case. It must be noted that initial point choice is crucial for the optimization process. Polak-Ribiere-Polyak version of CG is used below.





To compare robustness of the algorithms following method is suggested (related to the one used in [5]):

1. Generate set of points uniformly distributed in $\mathbf{G} = [x_{min}, x_{max}] \times [y_{min}, y_{max}]$:

$$P = \{p_i\}_{i=1}^N \in unif([x_{min}, x_{max}] \times [y_{min}, y_{max}])$$

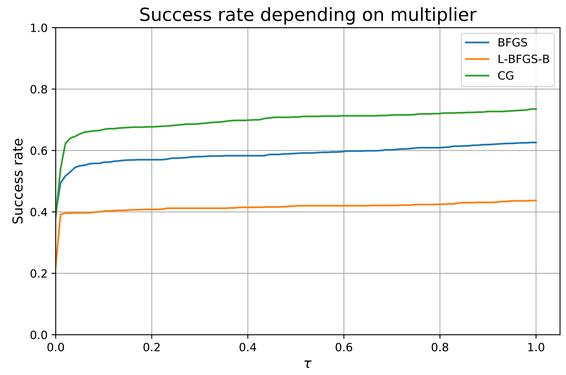
2. Take set $\mathcal{A} = \{A_1, A_2, \ldots\}$ of algorithms to be compared. Try to find the minimizer taking p_i as starting point and save the value obtained by algorithm A_k as $F_{A_k}(p_i)$. Denote $F_{min}(p_i) = \min_k F_{A_k}(p_i)$.
3. Take set of $\tau \in [0, +\infty]$ and compute [S]uccess rate as

$$S(A_k, \tau) = \frac{||\{p_i \mid F_{A_k}(p_i) \leq \tau F_{min}(p_i)\}||}{||P||}$$

4. Plot $S(A_k, \tau)$ for each $A_k \in \mathcal{A}$ as function of τ.

Results

Numerical experiments [7] were performed with $||P|| = 10000$ points for Beale function [6] and $p_i \in unif([-4.5, -4.5] \times [-4.5, -4.5])$. The plot shows that the **CG** algorithm performs at least as well as **BFGS** and **L-BFGS** methods. Moreover, while offering lower memory consumption (as **L-BFGS** does), it also has higher reliability. So we can do an informal statement: while the family of **BFGS** methods collects information about the curvature during optimization process, they explicitly assume that the function doesn't change too much between the steps. This also means that we can't move too far each iteration. In contrast, **CG** searches for a minimum along given direction and can perform quite big steps, while keeping the search directions very 'unique' in the sense of conjugacy.



References

[1] Fletcher, R. and Reeves, C., 1964, Function minimization by conjugate gradients. Computer Journal, 7, 149154.

[2] SciPy, Optimization module

[3] Dai, Y.H. and Yuan, Y., 2000, A nonlinear conjugate gradient method with a strong global convergence property. SIAM Journal on Optimization, 10, 177182.

[4] William W. Hager, Hongchao Zhang, A new conjugate gradient method with guaranteed descent and an efficient line search, SIAM J. Optim.16 (1) (2005) 170192.

[5] Andrei N. An adaptive conjugate gradient algorithm for large-scale unconstrained optimization. J Comput Appl Math, 2016, 292: 8391

[6] Surjanovic, S. & Bingham, D. (2013). Virtual Library of Simulation Experiments: Test Functions and Datasets.

[7] Google Colab