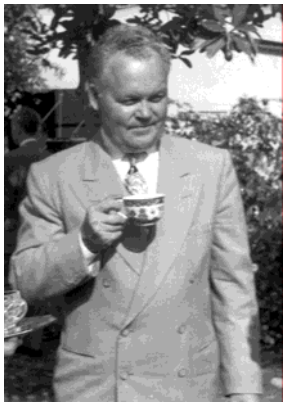
40 CONJUGATE GRADIENT ALGORITHMS FOR UNCONSTRAINED OPTIMIZATION
A survey on their definition

Neculai Andrei
Research Institute for Informatics,
Center for Advanced Modeling and Optimization,
8-10, Averescu Avenue, Bucharest 1, Romania,
and
Academy of Romanian Scientists,
54, Splaiul Independentei, Bucharest 5, Romania.
E-mail: nandrei@ici.ro

Abstract. In this work we present the conjugate gradient algorithms with special attention on their definition. 40 nonlinear conjugate gradient algorithms are presented. For each of them we present the formula for β_k definition or the main ingredients for algorithm definition. The conjugate gradient algorithms can be classified in 6 groups: classical, hybrid, modified, scaled, parametrized and accelerated.

Conjugate gradient algorithms are characterized by strong local and global convergence properties and low memory requirements. The history of these methods begins with the research of Magnus Hestenes at the Institute for Numerical Analysis and with independent work of Eduard Stiefel at the Technische Hochschule Zürich.


Magnus Hestenes (1906-1991)


Eduard Stiefel (1909-1978)

In this survey we focus on conjugate gradient methods for solving the nonlinear unconstrained optimization problem:

$$\min_{x \in R^n} f(x),$$

where $f : R^n \to R$ is a continuously differentiable function, bounded from below. Starting from an initial guess, a nonlinear conjugate gradient algorithm generates a sequence of points $\{x_k\}$, according to the following recurrence formula:

$$x_{k+1} = x_k + \alpha_k d_k,$$

where α_k is the step length, usually obtained by Wolfe line search,

$$f(x_k + \alpha_k d_k) - f(x_k) \le \rho \alpha_k g_k^T d_k,$$
$$g_{k+1}^T d_k \ge \sigma g_k^T d_k,$$

with $0 < \rho < 1/2 \le \sigma < 1,$ and the directions d_k are computed as:

$$d_{k+1} = -g_{k+1} + \beta_k s_k, \quad d_0 = -g_0.$$

Here β_k is a scalar known as the conjugate gradient parameter, $g_k = \nabla f(x_k)$ and $s_k = x_{k+1} - x_k$. In the following $y_k = g_{k+1} - g_k$. Different conjugate gradient algorithms correspond to different choices for the parameter β_k. Therefore, a crucial element in any conjugate gradient algorithm is the formula definition of β_k. Any conjugate gradient algorithm has a very simple general structure as it is illustrated below.

The prototype of Conjugate Gradient Algorithm

Step 1. Select the initial starting point $x_0 \in dom\, f$ and compute: $f_0 = f(x_0)$ and $g_0 = \nabla f(x_0)$. Set $d_0 = -g_0$ and $k = 0$.

Step 2. Test a criterion for stopping the iterations. For example, if $\|g_k\|_\infty \le \varepsilon$, then stop; otherwise continue with step 3.

Step 3. Using the Wolfe line search conditions:

$$f(x_k + \alpha_k d_k) - f(x_k) \le \rho \alpha_k g_k^T d_k,$$
$$g_{k+1}^T d_k \ge \sigma g_k^T d_k,$$

with $0 < \rho < 1/2 \le \sigma < 1,$ determine the steplength α_k.

Step 4. Update the variables as: $x_{k+1} = x_k + \alpha_k d_k$. Compute f_{k+1} and g_{k+1}. Compute $y_k = g_{k+1} - g_k$ and $s_k = x_{k+1} - x_k$.

Step 5. **Determine β_k.**

Step 6. Compute the search direction as: $d_{k+1} = -g_{k+1} + \beta_k s_k$.

Step 7. Restart criterion. If the restart criterion of Powell $\left|g_{k+1}^T g_k\right| > 0.2\|g_{k+1}\|^2$ is satisfied, then set $d_{k+1} = -g_{k+1}$.

Step 8. Compute the initial guess $\alpha_k = \alpha_{k-1}\|d_{k-1}\| / \|d_k\|$, set $k = k+1$ and continue with step 2. ■

1. Hestenes - Stiefel (HS)[1]

$$\beta_k^{HS} = \frac{y_k^T g_{k+1}}{y_k^T s_k}.$$

2. Fletcher - Reeves (FR)[2]

$$\beta_k^{FR} = \frac{g_{k+1}^T g_{k+1}}{g_k^T g_k}.$$

3. Polak – Ribière - Polyak (PRP)[3]

[1] **M.R. Hestenes** and **E.L. Stiefel,** *Methods of conjugate gradients for solving linear systems*, J. Research Nat. Bur. Standards, 49 (1952), pp.409-436.

[2] **R. Fletcher** and **C. Reeves,** *Function minimization by conjugate gradients*, Comput. J., 7 (1964), pp.149-154.

[3] **E. Polak** and **G. Ribière,** *Note sur la convergence de directions conjuguée*, Rev. Francaise Informat Recherche Operationelle, 3e Année 16 (1969), pp.35-43.
B.T. Polyak, *The conjugate gradient method in extreme problems*. USSR Comp. Math. Math. Phys., 9 (1969), pp.94-112.

$$\beta_k^{PRP} = \frac{y_k^T g_{k+1}}{g_k^T g_k}.$$

4. Polak – Ribière - Polyak plus (PRP+)[4]

$$\beta_k^{PRP} = \max\left\{0, \frac{y_k^T g_{k+1}}{g_k^T g_k}\right\}.$$

5. Conjugate Descent – Fletcher (CD)[5]

$$\beta_k^{CD} = -\frac{g_{k+1}^T g_{k+1}}{g_k^T d_k}.$$

6. Liu - Storey (LS)[6]

$$\beta_k^{LS} = -\frac{y_k^T g_{k+1}}{g_k^T d_k}.$$

7. Dai – Yuan (DY)[7]

$$\beta_k^{DY} = \frac{g_{k+1}^T g_{k+1}}{y_k^T s_k}.$$

8. Dai – Liao (DL)[8]

$$\beta_k^{DL} = \frac{g_{k+1}^T (y_k - t s_k)}{y_k^T s_k}.$$

9. Dai – Liao plus (DL+)[9]

$$\beta_k^{DL+} = \max\left\{0, \frac{y_k^T g_{k+1}}{y_k^T s_k}\right\} - t\frac{s_k^T g_{k+1}}{y_k^T s_k}.$$

10. Andrei - Sufficient Descent Condition (CGSD)[10]

$$\beta_k^{CGSD} = \frac{g_{k+1}^T g_{k+1}}{y_k^T s_k} - \frac{(y_k^T g_{k+1})(s_k^T g_{k+1})}{(y_k^T s_k)^2}.$$

11. Hybrid Dai - Yuan (hDY)[11]

---

[4] **M.J.D. Powell,** *Nonconvex minimization calculations and the conjugate gradient method.* Numerical Analysis (Dundee, 1983) Lecture Notes in mathematics, vol.1066, Springer-Verlag, Berlin, 1984, pp.122-141.

[5] **R. Fletcher,** *Practical Methods of Optimization, vol. 1: Unconstrained Optimization,* John Wiley & Sons, New York, 1987.

[6] **Y. Liu,** and **C. Storey,** *Efficient generalized conjugate gradient algorithms, Part 1: Theory.* JOTA, 69 (1991), pp.129-137.

[7] **Y.H. Dai** and **Y. Yuan,** *An efficient hybrid conjugate gradient method for unconstrained optimization,* Ann. Oper. Res., 103 (2001), pp. 33-47.

[8] **Y.H. Dai** and **L.Z. Liao,** *New conjugacy conditions and related nonlinear conjugate gradient methods.* Appl. Math. Optim., 43 (2001), pp. 87-101.

[9] **H. Yabe** and **M. Takano,** *Global convergence properties of nonlinear conjugate gradient methods with modified secant conditions.* Computational Optimization and Applications, 28 (2004), pp.203-225.

[10] **N. Andrei**, *A Dai-Yuan conjugate gradient algorithm with sufficient descent and conjugacy conditions for unconstrained optimization.* Applied Mathematics Letters, vol.21, 2008, pp.165-171.

[11] **Y.H. Dai** and **Y. Yuan,** *An efficient hybrid conjugate gradient method for unconstrained optimization,* Ann. Oper. Res., 103 (2001), pp. 33-47.

$$\beta_k^{hDY} = max\left\{c\beta^{DY}, min\left\{\beta^{HS}, \beta^{DY}\right\}\right\},$$
$$c = -(1-\sigma)/(1+\sigma).$$

12. Hybrid Dai – Yuan zero (hDYz)[12]

$$\beta_k^{hDYz} = max\left\{0, min\left\{\beta^{HS}, \beta^{DY}\right\}\right\}$$

13. Gilbert – Nocedal (GN)[13]

$$\beta_k^{GN} = max\left\{-\beta^{FR}, min\left\{\beta^{PRP}, \beta^{FR}\right\}\right\}$$

14. Hu – Storey (HuS)[14]

$$\beta_k^{HuS} = max\left\{0, min\left\{\beta^{PRP}, \beta^{FR}\right\}\right\}$$

15. Touati-Ahmed and Storey (TaS)[15]

$$\beta_k^{TaS} = \begin{cases} \beta^{PRP} & 0 \le \beta^{PRP} \le \beta^{FR}, \\ \beta^{FR} & \text{otherwise} \end{cases}$$

16. Hybrid LS – CD (LS-CD)

$$\beta_k^{LS-CD} = max\left\{0, min\left\{\beta^{LS}, \beta^{CD}\right\}\right\}$$

17. Birgin – Martínez (BM)[16]

$$\beta_k^{BM} = \frac{(\theta_k y_k - s_k)^T g_{k+1}}{y_k^T s_k},$$

where θ_k is the spectral gradient:

$$\theta_k = \frac{s_k^T s_k}{y_k^T s_k}.$$

18. Birgin – Martínez plus (BM+)

$$\beta_k^{BM+} = max\left\{0, \frac{\theta y_k^T g_{k+1}}{y_k^T s_k}\right\} - \frac{s_k^T g_{k+1}}{y_k^T s_k}.$$

where θ_k is the spectral gradient:

$$\theta_k = \frac{s_k^T s_k}{y_k^T s_k}.$$

---

[12] **Y.H. Dai** and **Y. Yuan,** *An efficient hybrid conjugate gradient method for unconstrained optimization,* Ann. Oper. Res., 103 (2001), pp. 33-47.
Y.H. Dai and **Q. Ni,** *Testing different conjugate gradient methods for large-scale unconstrained optimization.* J. Comput. Math., 21 (2003), pp.311-320.
[13] **J.C. Gilbert** and **J. Nocedal,** *Global convergence properties of conjugate gradient methods for optimization*, SIAM J. Optim., 2 (1992), pp. 21-42.
[14] **Y.F. Hu** and **C. Storey,** *Global convergence result for conjugate gradient methods.* J. Optim. Theory Appl., 71 (1991), pp.399-405.
[15] **D. Touati-Ahmed** and **C. Storey,** *Efficient hybrid conjugate gradient techniques.* J. Optim. Theory Appl., 64 (1990), pp.379-397.
[16] **E. Birgin** and **J.M. Martínez,** *A spectral conjugate gradient method for unconstrained optimization*, Applied Mathematics and Optimization, 43, pp.117-128, 2001.

19. Scaled Polak-Ribière-Polyak (sPRP)[17]

$$\beta_k^{sPRP} = \frac{\theta_{k+1} y_k^T g_{k+1}}{\alpha_k \theta_k g_k^T g_k},$$

where θ_k is the spectral gradient:

$$\theta_k = \frac{s_k^T s_k}{y_k^T s_k}.$$

20. Scaled Fletcher – Reeves (sFR)[18]

$$\beta_k^{sFR} = \frac{\theta_{k+1} g_{k+1}^T g_{k+1}}{\alpha_k \theta_k g_k^T g_k},$$

where θ_k is the spectral gradient:

$$\theta_k = \frac{s_k^T s_k}{y_k^T s_k}.$$

21. Scaled Hestenes – Stiefel (sHS)

$$\beta_k^{sHS} = \frac{s_k^T \nabla^2 f(x_{k+1}) g_{k+1}}{s_k^T \nabla^2 f(x_{k+1}) s_k}$$

22. Daniel (D)[19]

$$\beta_k^{D} = \frac{s_k^T \nabla^2 f(x_{k+1}) g_{k+1}}{s_k^T \nabla^2 f(x_{k+1}) s_k}$$

23. Andrei – Sufficient Descent Condition from PRP (A-prp)[20]

$$\beta_k^{A-prp} = \frac{1}{y_k^T s_k}\left(y_k^T g_{k+1} - \frac{(y_k^T y_k)(s_k^T g_{k+1})}{g_k^T g_k} \right).$$

24. Andrei – Sufficient Descent Condition from DY (ACGA)[21]

$$\beta_k^{ACGA} = \frac{y_k^T g_{k+1}}{y_k^T s_k} - \frac{(y_k^T g_{k+1})(s_k^T g_{k+1})}{(y_k^T s_k)^2}.$$

25. Andrei – Sufficient Descent Condition from DY zero (ACGA+)

$$\beta_k^{ACGA+} = max\left\{ 0, \frac{y_k^T g_{k+1}}{y_k^T s_k} \right\}\left(1 - \frac{s_k^T g_{k+1}}{y_k^T s_k} \right).$$

---

[17] **N. Andrei,** *Scaled memoryless BFGS preconditioned conjugate gradient algorithm for unconstrained optimization.* Optimization Methods and Software, vol. 22, no. 4, 2007, pp.561-571.

[18] **N. Andrei,** *Scaled conjugate gradient algorithms for unconstrained optimization*. Computational Optimization and Applications, vol. 38, no. 3, 2007, pp.401-416.

[19] **J.W. Daniel,** *The conjugate gradient method for linear and nonlinear operator equations.* SIAM J. Numer. Anal., 4 (1967), pp.10-26.

[20] **N. Andrei,** *Another nonlinear conjugate gradient algorithm with sufficient descent conditions for unconstrained optimization.* ICI Technical Reports, November 22, 2006

[21] **N. Andrei,** *Another nonlinear conjugate gradient algorithm for unconstrained optimization.* ICI Technical Report, May 16, 2007.

26. Convex combination of PRP and DY from conjugacy condition (CCOMB - Andrei)[22]

$$\beta_k^{CCOMB} = (1-\theta_k)\beta_k^{PRP} + \theta_k \beta_k^{DY},$$

where

$$\theta_k = \theta_k^{CCOMB} = \frac{(y_k^T g_{k+1})(y_k^T s_k) - (y_k^T g_{k+1})(g_k^T g_k)}{(y_k^T g_{k+1})(y_k^T s_k) - \|g_{k+1}\|^2 \|g_k\|^2}.$$

If $\theta_k^{CCOMB} \le 0$, then $\beta_k = \beta_k^{PRP}$. If $\theta_k^{CCOMB} \ge 1$, then $\beta_k = \beta_k^{DY}$.

27. Convex combination of PRP and DY from Newton direction (NDOMB - Andrei)[23]

$$\beta_k^{NDOMB} = (1-\theta_k)\beta_k^{PRP} + \theta_k \beta_k^{DY},$$

where

$$\theta_k = \theta_k^{NDOMB} = \frac{(y_k^T g_{k+1} - s_k^T g_{k+1})\|g_k\|^2 - (g_{k+1}^T y_k)(y_k^T s_k)}{\|g_{k+1}\|^2 \|g_k\|^2 - (g_{k+1}^T y_k)(y_k^T s_k)}.$$

If $\theta_k^{NDOMB} \le 0$, then $\beta_k^{NDOMB} = \beta_k^{PRP}$. If $\theta_k^{NDOMB} \ge 1$, then $\beta_k = \beta_k^{DY}$.

28. Convex combination of HS and DY from Newton direction (HYBRID - Andrei)[24]

$$\beta_k^{HYBRID} = (1-\theta_k)\beta_k^{HS} + \theta_k \beta_k^{DY},$$

where

$$\theta_k = -\frac{s_k^T g_{k+1}}{g_k^T g_{k+1}}.$$

If $\theta_k \le 0$, then $\beta_k^{HYBRID} = \beta_k^{HS}$. If $\theta_k \ge 1$, then $\beta_k^{HYBRID} = \beta_k^{DY}$.

29. Convex combination of HS and DY from Newton direction with modified secant condition (HYBRIDM - Andrei)[25]

$$\beta_k^{HYBRIDM} = (1-\theta_k)\beta_k^{HS} + \theta_k \beta_k^{DY},$$

where

$$\theta_k = \frac{\left(\frac{\delta\eta_k}{s_k^T s_k} - 1\right) s_k^T g_{k+1} - \frac{y_k^T g_{k+1}}{y_k^T s_k}\delta\eta_k}{g_k^T g_{k+1} + \frac{g_k^T g_{k+1}}{y_k^T s_k}\delta\eta_k}.$$

---

[22] **N. Andrei,** *New hybrid conjugate gradient algorithms as a convex combination of PRP and DY for unconstrained optimization.* ICI Technical Report, October 1, 2007.

[23] **N. Andrei,** *New hybrid conjugate gradient algorithms for unconstrained optimization.* Encyclopedia of Optimization, 2nd Edition, C.A. Floudas, and P. Pardalos (Eds.), August 2008, Entry 761.

N. Andrei, *New hybrid conjugate gradient algorithms for unconstrained optimization*. Encyclopedia of Optimization, 2nd Edition, C.A. Floudas and P. Pardalos (Eds.) August 2008, Entry 761.

[24] **N. Andrei,** *Another conjugate gradient algorithm for unconstrained optimization.* ICI Technical Report, August 12, 2007.

N. Andrei, *A hybrid conjugate gradient algorithm for unconstrained optimization as a convex combination of Hestenes-Stiefel and Dai-Yuan.* Studies in Informatics and Control, vol.17, No.1, March 2008, pp.55-70.

[25] **N. Andrei,** *A hybrid conjugate gradient algorithm with modified secant condition for unconstrained optimization.* ICI Technical Report, February 6, 2008

δ is a parameter. For $\delta = 0$ we get HYBRID. If $\theta_k \le 0$, then $\beta_k^{HYBRIDM} = \beta_k^{HS}$. If $\theta_k \ge 1$, then $\beta_k^{HYBRIDM} = \beta_k^{DY}$.

30. Guaranteed descent with efficient line search (CG_DESCENT - Hager and Zhang)[26]

$$d_{k+1} = -g_{k+1} + \bar{\beta}_k^{HZ} d_k, \quad d_0 = -g_0,$$

$$\bar{\beta}_k^{HZ} = max\left\{\beta_k^{HZ}, \eta_k\right\},$$

$$\eta_k = \frac{-1}{\|d_k\| min\left\{\eta, \|g_k\|\right\}}, \qquad \eta = 0.01$$

$$\beta_k^{HZ} = \frac{1}{y_k^T d_k}\left(y_k - 2\frac{\|y_k\|^2}{y_k^T d_k} d_k\right)^T g_{k+1}$$

31. Yabe – Takano (YT)[27]

$$\beta_k^{YT} = \frac{g_{k+1}^T (z_k - ts_k)}{d_k^T z_k},$$

where $z_k = y_k + \frac{\delta \xi_k}{s_k^T u_k} u_k$, $\xi_k = 6(f_k - f_{k+1}) + 3(g_k + g_{k+1})^T s_k$, $\delta \ge 0$ is a constant and $u_k \in R^n$ satisfies $s_k^T u_k \ne 0$; for example $u_k = s_k$.

32. Yabe – Takano plus (YT+)[28]

$$\beta_k^{YT+} = \max\left\{0, \frac{g_{k+1}^T z_k}{d_k^T z_k}\right\} - t\frac{g_{k+1}^T s_k}{d_k^T z_k}.$$

where $z_k = y_k + \frac{\delta \xi_k}{s_k^T u_k} u_k$, $\xi_k = 6(f_k - f_{k+1}) + 3(g_k + g_{k+1})^T s_k$, $\delta \ge 0$ is a constant and $u_k \in R^n$ satisfies $s_k^T u_k \ne 0$; for example $u_k = s_k$.

33. BFGS preconditioned (CONMIN – Shanno and Phua)[29]

---

[26] **W.W. Hager,** and **H. Zhang,** *A new conjugate gradient method with guaranteed descent and an efficient line search*. SIAM Journal on Optimization, 16 (2005) 170-192.
W.W. Hager, and **H. Zhang,** *Algorithm 851: CG_DESCENT, A conjugate gradient method with guaranteed descent*. ACM Transactions on Mathematical Software, 32 (2006), 113-137.

[27] **H. Yabe, H.** and **M. Takano,** *Global convergence properties of nonlinear conjugate gradient methods with modified secant conditions.* Computational Optimization and Applications, (COAP) 28, (2004), pp.203-225.

[28] **H. Yabe, H.** and **M. Takano,** *Global convergence properties of nonlinear conjugate gradient methods with modified secant conditions.* Computational Optimization and Applications, (COAP) 28, (2004), pp.203-225.

[29] **D.F. Shanno,** *Conjugate gradient methods with inexact searches*. Mathematics of Operations Research, vol. 3 (1978), pp.244-256.
D.F. Shanno, *On the convergence of a new conjugate gradient algorithm.* SIAM J. Numer. Anal., 15 (1978), pp.1247-1257.
D.F. Shanno, and **K.H. Phua,** *Algorithm 500, Minimization of unconstrained multivariate functions*, ACM Trans. on Math. Software, 2 (1976) 87-94.
D.F. Shanno, and **K.H. Phua,** *Remark on algorithm 500.* ACM Trans. on Math. Software, 6 (1980), pp. 618-622.

Step 1. Initialization. Select $x_0 \in R^n$, and the parameters $0 < \rho < 1/2 \le \sigma < 1$. Compute $f(x_0)$ and $g_0 = \nabla f(x_0)$. Set $d_0 = -g_0$ and $\alpha_0 = 1/\|g_0\|$. Set $k = 0$.

Step 2. Compute the initial guess: $\alpha_k = \alpha_{k-1}\|d_{k-1}\|_2 / \|d_k\|_2$. Using this initialization compute α_k satisfying the Wolfe conditions.

Step 3. Update the variables: $x_{k+1} = x_k + \alpha_k d_k$. Compute $f(x_{k+1})$, g_{k+1} and $s_k = x_{k+1} - x_k$, $y_k = g_{k+1} - g_k$.

Step 4. Test a criterion for stopping the iterations. For example, if $\|g_k\|_\infty \le \varepsilon$, then stop; otherwise continue with step 5.

Step 5. Test for restart. If the iteration k is a multiple of n, or $|g_{k+1}^T g_k| \ge 0.2\|g_{k+1}\|^2$, then compute d_{k+1} as:

$$d_{k+1} = -\gamma g_{k+1} - \left[\left(1 - \gamma\frac{y_k^T y_k}{y_k^T s_k}\right)\frac{s_k^T g_{k+1}}{y_k^T s_k} - \gamma\frac{y_k^T g_{k+1}}{y_k^T s_k}\right]s_k + \gamma\frac{s_k^T g_{k+1}}{y_k^T s_k}y_k .$$

Consider: $s_t = d_k$, $y_t = y_k$, $k = k+1$ and continue with step 2. Otherwise, continue with step 6.

Step 6. Compute:

$$d_{k+1} = -\hat{H}_k g_{k+1} + \frac{s_k^T g_{k+1}}{y_k^T s_k}\hat{H}_k y_k - \left[\left(1 + \frac{y_k^T \hat{H}_k y_k}{y_k^T s_k}\right)\frac{s_k^T g_{k+1}}{y_k^T s_k} - \frac{y_k^T \hat{H}_k g_{k+1}}{y_k^T s_k}\right]s_k ,$$

where the vectors $\hat{H}_k g_{k+1}$ and $\hat{H}_k y_k$ are computed as:

$$\hat{H}_k g_{k+1} = \frac{y_t^T s_t}{y_t^T y_t}g_{k+1} - \frac{s_t^T g_{k+1}}{y_t^T y_t}y_t + \left[2\frac{s_t^T g_{k+1}}{y_t^T s_t} - \frac{y_t^T g_{k+1}}{y_t^T y_t}\right]s_t ,$$

$$\hat{H}_k y_k = \frac{y_t^T s_t}{y_t^T y_t}y_k - \frac{s_t^T y_k}{y_t^T y_t}y_t + \left[2\frac{s_t^T y_k}{y_t^T s_t} - \frac{y_t^T y_k}{y_t^T y_t}\right]s_t .$$

Step 7. Scale de direction: d_{k+1} as:

$$d_{k+1} = [2(f(x_{k+1}) - f(x_k))/d_{k+1}^T g_{k+1}]d_{k+1} ,$$

set $k = k+1$ and continue with step 2. ■

34. Scaled BFGS preconditioned (SCALCG - Andrei)[30]

Step 1. Initialization. Select $x_0 \in R^n$, and the parameters $0 < \rho < 1/2 \le \sigma < 1$. Compute $f(x_0)$ and $g_0 = \nabla f(x_0)$. Set $d_0 = -g_0$ and $\alpha_0 = 1/\|g_0\|$. Set $k = 0$.

[30] **N. Andrei,** *Scaled memoryless BFGS preconditioned conjugate gradient algorithm for unconstrained optimization.* Optimization Methods and Software, 22 (2007), pp.561-571.
N. Andrei, *A scaled BFGS preconditioned conjugate gradient algorithm for unconstrained optimization.* Applied Mathematics Letters, 20 (2007), pp.645-650.
N. Andrei, *Scaled conjugate gradient algorithms for unconstrained optimization.* Computational Optimization and Applications, vol. 38, no. 3, 2007, pp.401-416.
N. Andrei. *A scaled nonlinear conjugate gradient algorithm for unconstrained optimization.* Optimization. A journal of mathematical programming and operations research. Accepted, *In press.*
N. Andrei, - *SCALCG – Scaled Conjugate Gradient Algorithms for Unconstrained Optimization.* Technical Report ICI, No. 17/2007. March 30, 2007. (CD included)

Step 2. Line search. Compute α_k satisfying the Wolfe conditions. Update the variables $x_{k+1} = x_k + \alpha_k d_k$. Compute $f(x_{k+1}), g_{k+1}$ and $s_k = x_{k+1} - x_k$, $y_k = g_{k+1} - g_k$.
Step 3. Test for continuation of iterations. If this test is satisfied the iterations are stopped, else set $k = k+1$.

Step 4. Scaling factor computation. Compute $\theta_{k+1} = \dfrac{s_k^T s_k}{y_k^T s_k}$.
Step 5. Restart direction. Compute the (restart) direction d_k as:

$$d_{k+1} = -\theta_{k+1} g_{k+1} + \theta_{k+1}\left(\frac{g_{k+1}^T s_k}{y_k^T s_k}\right) y_k - \left[\left(1+\theta_{k+1}\frac{y_k^T y_k}{y_k^T s_k}\right)\frac{g_{k+1}^T s_k}{y_k^T s_k} - \theta_{k+1}\frac{g_{k+1}^T y_k}{y_k^T s_k}\right] s_k .$$

Step 6. Line search. Compute the initial guess: $\alpha_k = \alpha_{k-1}\|d_{k-1}\|_2 / \|d_k\|_2$. Using this initialization compute α_k satisfying the Wolfe conditions. Update the variables $x_{k+1} = x_k + \alpha_k d_k$. Compute $f(x_{k+1})$, g_{k+1} and $s_k = x_{k+1} - x_k$, $y_k = g_{k+1} - g_k$.
Step 7. Store: $\theta = \theta_k$, $s = s_k$ and $y = y_k$.
Step 8. Test for continuation of iterations. If this test is satisfied the iterations are stopped, else set $k = k+1$.

Step 9. Restart. If the Powell restart criterion: $\left|g_{k+1}^T g_k\right| \geq 0.2\|g_{k+1}\|^2$, is satisfied, then go to step 4 (a restart step); otherwise continue with step 10 (a standard step).
Step 10. Standard direction. Compute the direction d_k as:

$$d_{k+1} = -v + \frac{(g_{k+1}^T s_k)w + (g_{k+1}^T w)s_k}{y_k^T s_k} - \left(1+\frac{y_k^T w}{y_k^T s_k}\right)\frac{g_{k+1}^T s_k}{y_k^T s_k} s_k ,$$

where v and w are computed as:

$$v = \theta g_{k+1} - \theta\left(\frac{g_{k+1}^T s}{y^T s}\right) y + \left[\left(1+\theta\frac{y^T y}{y^T s}\right)\frac{g_{k+1}^T s}{y^T s} - \theta\frac{g_{k+1}^T y}{y^T s}\right] s ,$$

and

$$w = \theta\, y_k - \theta\left(\frac{y_k^T s}{y^T s}\right) y + \left[\left(1+\theta\frac{y^T y}{y^T s}\right)\frac{y_k^T s}{y^T s} - \theta\frac{y_k^T y}{y^T s}\right] s ,$$

with saved values θ, s and y.

Step 11. Line search. Compute the initial guess: $\alpha_k = \alpha_{k-1}\|d_{k-1}\|_2 / \|d_k\|_2$. Using this initialization compute α_k satisfying the Wolfe conditions. Update the variables $x_{k+1} = x_k + \alpha_k d_k$. Compute $f(x_{k+1})$, g_{k+1} and $s_k = x_{k+1} - x_k$, $y_k = g_{k+1} - g_k$.
Step 12. Test for continuation of iterations. If this test is satisfied the iterations are stopped, else set $k = k+1$ and go to step 9. ■

Remark:

In *Step 4,* θ_{k+1} can be computed in an anticipative way as: $\theta_{k+1} = \dfrac{1}{\gamma_{k+1}}$, where

$$\gamma_{k+1} = \frac{2}{d_k^T d_k}\frac{1}{\alpha_k^2}\left[f(x_{k+1}) - f(x_k) - \alpha_k g_k^T d_k\right].$$

Observe that $\gamma_{k+1} > 0$ for convex functions. If $f(x_{k+1}) - f(x_k) - \alpha_k g_k^T d_k < 0$, then the reduction $f(x_{k+1}) - f(x_k)$ in function value is smaller than $\alpha_k g_k^T d_k$. In these cases the idea is to reduce a little the step size α_k as $\alpha_k - \eta_k$, maintaining the other quantities at their

values in such a way so that γ_{k+1} is positive. To get a value for η_k let us select a real $\delta > 0$, " small enough" but comparable with the value of the function, and have

$$\eta_k = \frac{1}{g_k^T d_k}\left[f(x_k) - f(x_{k+1}) + \alpha_k g_k^T d_k + \delta \right],$$

with which a new value for γ_{k+1} can be computed as:

$$\gamma_{k+1} = \frac{2}{d_k^T d_k}\frac{1}{(\alpha_k - \eta_k)^2}\left[f(x_{k+1}) - f(x_k) - (\alpha_k - \eta_k) g_k^T d_k \right].$$

35. Accelerated scaled BFGS preconditioned (ASCALCG - Andrei)[31]

Step 1. Initialization. Select $x_0 \in R^n$, and the parameters $0 < \sigma_1 \le \sigma_2 < 1$. Compute $f(x_0)$ and $g_0 = \nabla f(x_0)$. Set $d_0 = -g_0$ and $\alpha_0 = 1/\|g_0\|$. Set $k = 0$.

Step 2. Line search. Compute α_k satisfying the Wolfe conditions. Update the variables $x_{k+1} = x_k + \alpha_k d_k$. Compute $f(x_{k+1}), g_{k+1}$ and $s_k = x_{k+1} - x_k$, $y_k = g_{k+1} - g_k$.

Step 3. Test for continuation of iterations. If this test is satisfied the iterations are stopped, else set $k = k+1$.

Step 4. Scaling factor computation. Compute $\theta_{k+1} = \dfrac{s_k^T s_k}{y_k^T s_k}$.

Step 5. Restart direction. Compute the (restart) direction d_k as:

$$d_{k+1} = -\theta_{k+1} g_{k+1} + \theta_{k+1}\left(\frac{g_{k+1}^T s_k}{y_k^T s_k}\right) y_k - \left[\left(1 + \theta_{k+1}\frac{y_k^T y_k}{y_k^T s_k}\right)\frac{g_{k+1}^T s_k}{y_k^T s_k} - \theta_{k+1}\frac{g_{k+1}^T y_k}{y_k^T s_k}\right] s_k.$$

Step 6. Line search. Compute the initial guess: $\alpha_k = \alpha_{k-1}\|d_{k-1}\|_2 / \|d_k\|_2$. Using this initialization compute α_k satisfying the Wolfe conditions. Update the variables $x_{k+1} = x_k + \alpha_k d_k$. Compute $f(x_{k+1})$ and g_{k+1}.

Step 7. Acceleration scheme. Compute $a_k = g_k^T d_k$ and $b_k = (g_k - g_{k+1})^T d_k$. If $b_k \ne 0$, then compute $\gamma_k = a_k / b_k$ and update the variables as: $x_{k+1} = x_k + \gamma_k \alpha_k d_k$. Compute $f(x_{k+1})$, g_{k+1} and $s_k = x_{k+1} - x_k$, $y_k = g_{k+1} - g_k$. Otherwise (if $b_k = 0$), then compute $s_k = x_{k+1} - x_k$, $y_k = g_{k+1} - g_k$.

Step 8. Store: $\theta = \theta_k$, $s = s_k$ and $y = y_k$.

Step 9. Test for continuation of iterations. If this test is satisfied the iterations are stopped, else set $k = k+1$.

Step 10. Restart. If the Powell restart criterion: $\left|g_{k+1}^T g_k\right| \ge 0.2\|g_{k+1}\|^2$, is satisfied, then go to step 4 (a restart step); otherwise continue with step 11 (a standard step).

Step 11. Standard direction. Compute the direction d_k as:

$$d_{k+1} = -v + \frac{(g_{k+1}^T s_k) w + (g_{k+1}^T w) s_k}{y_k^T s_k} - \left(1 + \frac{y_k^T w}{y_k^T s_k}\right)\frac{g_{k+1}^T s_k}{y_k^T s_k} s_k,$$

where v and w are computed as:

[31] **N. Andrei,** *Accelerated scaled memoryless BFGS preconditioned conjugate gradient algorithm for unconstrained optimization.* ICI Technical Report, March 30, 2008

$$v = \theta g_{k+1} - \theta\left(\frac{g_{k+1}^T s}{y^T s}\right) y + \left[\left(1+\theta\frac{y^T y}{y^T s}\right)\frac{g_{k+1}^T s}{y^T s} - \theta\,\frac{g_{k+1}^T y}{y^T s}\right] s\ ,$$

and

$$w = \theta\, y_k - \theta\left(\frac{y_k^T s}{y^T s}\right) y + \left[\left(1+\theta\,\frac{y^T y}{y^T s}\right)\frac{y_k^T s}{y^T s} - \theta\,\frac{y_k^T y}{y^T s}\right] s\ ,$$

with saved values θ, s and y.

Step 12. Line search. Compute the initial guess: $\alpha_k = \alpha_{k-1}\|d_{k-1}\|_2 / \|d_k\|_2$. Using this initialization compute α_k satisfying the Wolfe conditions. Update the variables $x_{k+1} = x_k + \alpha_k d_k$. Compute $f(x_{k+1})$ and g_{k+1}.

Step 13. Acceleration scheme. Compute $a_k = g_k^T d_k$ and $b_k = (g_k - g_{k+1})^T d_k$. If $b_k \neq 0$, then compute $\gamma_k = a_k / b_k$ and update the variables as: $x_{k+1} = x_k + \gamma_k \alpha_k d_k$. Compute $f(x_{k+1})$, g_{k+1} and $s_k = x_{k+1} - x_k$, $y_k = g_{k+1} - g_k$. Otherwise (if $b_k = 0$), then compute $s_k = x_{k+1} - x_k$, $y_k = g_{k+1} - g_k$.

Step 14. Test for continuation of iterations. If this test is satisfied the iterations are stopped, else set $k = k+1$ and go to step 10. ■

36. Accelerated conjugate gradient algorithm from Newton direction with modified secant condition (ACGMSEC - Andrei)[32]

Step 1. Initialization. Select the initial starting point $x_0 \in dom\, f$ and compute: $f_0 = f(x_0)$ and $g_0 = \nabla f(x_0)$. Set $d_0 = -g_0$ and $k = 0$. Select a value for parameters ε and τ.

Step 2. Test a criterion for stopping the iterations. For example, if $\|g_k\|_\infty \le \varepsilon$, then stop; otherwise continue with step 3.

Step 3. Line search. Using the Wolfe line search conditions determine the steplength α_k.

Step 4. Compute: $z = x_k + \alpha_k d_k$, $g_z = \nabla f(z)$ and $y_k = g_k - g_z$.

Step 5. Compute: $a_k = g_k^T d_k$, and $b_k = y_k^T d_k$.

Step 6. Acceleration scheme. If $b_k \neq 0$, then compute $\gamma_k = a_k / b_k$ and update the variables as $x_{k+1} = x_k + \gamma_k \alpha_k d_k$, otherwise update the variables as $x_{k+1} = x_k + \alpha_k d_k$. Compute f_{k+1} and g_{k+1}. Compute $y_k = g_{k+1} - g_k$ and $s_k = x_{k+1} - x_k$.

Step 7. Set $\delta = 0$. If $\|s_k\| \le \tau$, then set $\delta = 1$.

Step 8. Determine β_k as:

$$\beta_k = \max\left\{\frac{y_k^T g_{k+1}}{y_k^T s_k + \delta\eta_k}, 0\right\} - \left(1 - \frac{\delta\eta_k}{\|s_k\|^2}\right)\frac{s_k^T g_{k+1}}{y_k^T s_k + \delta\eta_k}$$

Step 9. Direction computation. Compute the search direction as: $d_{k+1} = -g_{k+1} + \beta_k s_k$.

Step 10. Restart criterion. If the restart criterion of Powell $\left|g_{k+1}^T g_k\right| > 0.2\|g_{k+1}\|^2$ is satisfied, then set $d_{k+1} = -g_{k+1}$.

[32] **N. Andrei,** *Accelerated conjugate gradient algorithm with modified secant condition for unconstrained optimization.* ICI Technical Report, March 3, 2008.
N. Andrei, *Acceleration of conjugate gradient algorithms for unconstrained optimization.* ICI Technical Report, October 24, 2007 (submitted).

Step 11. Compute the initial guess $\alpha_k = \alpha_{k-1}\|d_{k-1}\| / \|d_k\|$, set $k = k+1$ and continue with step 2. ■

37. Accelerated conjugate gradient algorithm from Newton direction with finite difference HESSIAN / vector product approximation (ACGHES - Andrei)[33]

Step 1. Initialization. Select the initial starting point $x_0 \in dom\, f$ and compute: $f_0 = f(x_0)$ and $g_0 = \nabla f(x_0)$. Set $d_0 = -g_0$ and $k = 0$. Select a value for the parameter ε.

Step 2. Test a criterion for stopping the iterations. For example, if $\|g_k\|_\infty \le \varepsilon$, then stop; otherwise continue with step 3.

Step 3. Line search. Using the Wolfe line search conditions determine the steplength α_k.

Step 4. Compute: $z = x_k + \alpha_k d_k$, $g_z = \nabla f(z)$ and $y_k = g_k - g_z$.

Step 5. Compute: $a_k = g_k^T d_k$, and $b_k = y_k^T d_k$.

Step 6. Acceleration scheme. If $b_k \ne 0$, then compute $\gamma_k = a_k / b_k$ and update the variables as $x_{k+1} = x_k + \gamma_k \alpha_k d_k$, otherwise update the variables as $x_{k+1} = x_k + \alpha_k d_k$. Compute f_{k+1} and g_{k+1}. Compute $s_k = x_{k+1} - x_k$.

Step 7. Hessian / vector product approximation. Compute $\delta = \dfrac{2\sqrt{\varepsilon_m}(1+\|x_{k+1}\|)}{\|s_k\|}$ and $y_k = (\nabla f(x_{k+1} + \delta s_k) - \nabla f(x_{k+1}))/\delta$.

Step 8. Compute $\beta_k = (y_k^T g_{k+1} - s_k^T g_{k+1}) / s_k^T y_k$.

Step 9. Direction computation. Compute the search direction as $d_{k+1} = -g_{k+1} + \beta_k s_k$.

Step 10. Restart criterion. If the restart criterion of Powell $\left|g_{k+1}^T g_k\right| > 0.2\|g_{k+1}\|^2$ is satisfied, then set $d_{k+1} = -g_{k+1}$.

Step 11. Compute the initial guess $\alpha_k = \alpha_{k-1}\|d_{k-1}\| / \|d_k\|$, set $k = k+1$ and continue with step 2. ■

Remark:
In step 7 the computation of δ is implemented as:

$$\delta = \max\left\{\frac{\varphi}{\max\{10\varphi, \|s_k\|\}}, \frac{\varphi}{100}\right\}, \quad \varphi = 2\sqrt{\varepsilon_m}(1+\|x_{k+1}\|\sqrt{n}).$$

38. Parametrized CG with one parameter[34]

$$\beta_k = \frac{\|g_{k+1}\|^2}{\lambda_k \|g_k\|^2 + (1-\lambda_k) d_k^T y_k}, \ \lambda_k \in [0,1].$$

The FR algorithm corresponds to $\lambda_k = 1$. The DY algorithm correspond to $\lambda_k = 0$.

[33] **N. Andrei,** Accelerated conjugate gradient algorithm with finite difference Hessian / vector product approximation for unconstrained optimization. ICI Technical Report, March 4, 2008

[34] **Y.H., Dai, Y. Yuan**, *A three-parameter family of hybrid conjugate gradient method.* Mathematics of Computation, 70 (2001) pp.1155-1167.

39. Parametrized CG with two parameters[35]

$$\beta_k = \frac{\mu_k \|g_{k+1}\|^2 + (1-\mu_k) g_{k+1}^T y_k}{\lambda_k \|g_k\|^2 + (1-\lambda_k) d_k^T y_k}, \ \lambda_k, \mu_k \in [0,1].$$

This two parameter family includes the methods: FR, DY, PRP and HS in extreme cases.

40. Parametrized CG with three parameters[36]

$$\beta_k = \frac{\mu_k \|g_{k+1}\|^2 + (1-\mu_k) g_{k+1}^T y_k}{(1-\lambda_k-\omega_k)\|g_k\|^2 + \lambda_k d_k^T y_k - \omega_k d_k^T g_k},$$

$$\lambda_k, \mu_k \in [0,1] \ \text{ and } \ \omega_k \in [0, 1-\lambda_k].$$

This three parameter family includes the six classical conjugate gradient algorithms, as well as the previous one-parameter and two-parameter families.

March 14, 2008

--------oooooooOoooooooo--------

[35] **J.L. Nazareth.,** *Conjugate gradient methods.* Encyclopedia of Optimization, C.Floudas and P. Pardalos, (Eds.) Kluwer Academic Publishers, Boston, 1999.

[36] **Y.H., Dai, Y. Yuan**, *A three-parameter family of hybrid conjugate gradient method.* Mathematics of Computation, 70 (2001) pp.1155-1167.