Variants of the conjugate gradient algorithm with guaranteed descent and conjugacy conditions and a modified Wolfe line search DESCON for large-scale unconstrained optimization

Neculai Andrei
Center for Advanced Modeling and Optimization,
Academy of Romanian Scientists,
54, Splaiul Independenţei, Sector 5,
Bucharest, ROMANIA
E-mail: neculaiandrei70@gmail.com

Technical Report 1/2021
January 4, 2021

1. Variants of the algorithm DESCON

For solving the unconstrained optimization problems

$$\min_{x\in R^n} f(x), \tag{1}$$

where $f:\mathbb{R}^n \to \mathbb{R}$ is a continuously differentiable function, bounded from below, one of the most elegant and probably the most efficient method is the conjugate gradient method DESCON [1, 2, 3]. The algorithm is defined as:

$$x_{k+1} = x_k + \alpha_k d_k, \tag{2}$$

where $\alpha_k > 0$ is obtained by the Wolfe line search [4, 5]:

$$f(x_k + \alpha_k d_k) - f(x_k) \le \rho \alpha_k g_k^T d_k, \tag{3}$$

$$g(x_k + \alpha_k d_k)^T d_k \ge \sigma g_k^T d_k, \tag{4}$$

where $0 < \rho \le 1/2 < \sigma < 1,$ and d_k is supposed to be a descent direction, generated as:

$$d_{k+1} = -\theta_k g_{k+1} + \beta_k s_k, \tag{5}$$

$$\beta_k = \frac{y_k^T g_{k+1} - t_k s_k^T g_{k+1}}{y_k^T s_k}, \tag{6}$$

$d_0 = -g_0$, where θ_k and t_k are scalar parameters determined in such a way that both the descent and the conjugacy conditions are satisfied:

$$\theta_k = \frac{a_k - t_k (s_k^T g_{k+1})}{y_k^T g_{k+1}}, \tag{7}$$

$$t_k = \frac{b_k (y_k^T g_{k+1}) - a_k (y_k^T s_k) \|g_{k+1}\|^2}{\Delta_k}, \tag{8}$$

where:

$$\bar{\Delta}_k = (y_k^T g_{k+1})(s_k^T g_{k+1}) - \|g_{k+1}\|^2 (y_k^T s_k), \tag{9}$$

$$\Delta_k = (s_k^T g_{k+1}) \bar{\Delta}_k, \tag{10}$$

$$a_k = v(s_k^T g_{k+1}) + y_k^T g_{k+1}, \tag{11}$$

$$b_k = w\|g_{k+1}\|^2 (y_k^T s_k) + (y_k^T g_{k+1})(s_k^T g_{k+1}). \tag{12}$$

In DESCON a small modification of the second Wolfe line search condition (4) is used as:

$$g(x_k + \alpha_k d_k)^T d_k \geq \sigma_k g_k^T d_k, \tag{13}$$

where σ_k is a sequence of parameters satisfying the condition $0 < \rho < \sigma_k < 1$, for all k. Therefore, in DESCON the rate of decrease of f in the direction d_k at x_{k+1} is larger than a fraction σ_k, which is modified at every iteration, of the rate of decrease of f in the direction d_k at x_k. The condition $\rho < \sigma_k$, for all $k \geq 0$, guarantees that (3) and (13) can be satisfied simultaneously. Relations (3) and (13) are known as *the modified Wolfe conditions.*

The properties and the convergence of the algorithm are presented in [1, 2, 3]. Intensive numerical experiments showed that DESCON is more efficient and more robust versus some conjugate gradient methods, including here CG_DESCENT [6].

In this technical report we consider three new variants of DESCON subject to the computation of the conjugate gradient parameter β_k.

1) In the *first variant*, we call DESCON, the formula for computation the parameter β_k is that obtained from the sufficient descent and conjugacy conditions. Using (6) and the corresponding relations, in this variant the parameter β_k is computed as:

$$\beta_k = \frac{y_k^T g_{k+1}}{y_k^T s_k} - \frac{s_k^T g_{k+1}}{y_k^T s_k} \frac{b_k (y_k^T g_{k+1})}{\Delta_k} + \frac{(s_k^T g_{k+1}) a_k \|g_{k+1}\|^2}{\Delta_k}. \tag{14}$$

2) Having in view the PRP+ method introduced by Powell [7] in order to ensure the global convergence of the algorithms for general nonlinear function in the *second variant* of our algorithm, we call DESCON-P, the parameter β_k is computed as:

$$\tilde{\beta}_k = \min\left\{\frac{y_k^T g_{k+1}}{y_k^T s_k}, 0\right\} - \frac{s_k^T g_{k+1}}{y_k^T s_k} \frac{b_k (y_k^T g_{k+1})}{\Delta_k} + \frac{(s_k^T g_{k+1}) a_k \|g_{k+1}\|^2}{\Delta_k}. \tag{15}$$

The global convergence of this variant of DESCON can be proved using the same argument as in Dai and Liao [8].

3) In the *third variant* of our algorithm, we call DESCON-H, a truncation technique is introduced as suggested by Hager and Zhang [6]:

$$\bar{\beta}_k = \max\left(\beta_k, \frac{-1}{\|d_k\| \min(0.1, \|g_{k+1}\|)} \right), \tag{16}$$

where β_k is given by (14). Observe that in this computational scheme the lower bound on $\bar{\beta}_k$ is dynamically adjusted in order to make the lower bound smaller as iterates converge.

Numerical experiments

In the following we report some numerical results obtained with an implementation of the DESCON algorithm. The code is written in Fortran and compiled with f77 (default compiler settings) on a Workstation Intel Pentium 4 with 1.8 GHz. DESCON uses the loop unrolling to a depth of 5. We selected a number of 100 large-scale unconstrained optimization test functions in generalized or extended form [1] (some from CUTE library [9]). For each test function we have taken ten numerical experiments with the number of variables increasing as $n = 1000, 2000, ..., 10000.$ Therefore, all in all there are 1000 numerical experiments. The algorithm implements the Wolfe line search conditions with $\rho = 0.0001,$ $\sigma = \|g_{k+1}\|^2 / (|y_k^T g_{k+1}| + \|g_{k+1}\|^2),$ and the same stopping criterion $\|g_k\|_\infty \le 10^{-6}$, where $\|.\|_\infty$ is the maximum absolute component of a vector. In DESCON we set $w = 7/8$ and $v = 0.05$. In our numerical experiments θ_k is not restricted in the interval $[0, 2w].$ In all the algorithms we considered in this numerical study the maximum number of iterations is limited to 2000.

The comparisons of algorithms are given in the following context. Let f_i^{ALG1} and f_i^{ALG2} be the optimal value found by ALG1 and ALG2, for problem $i = 1, \ldots, 1000,$ respectively. We say that, in the particular problem $i,$ the performance of ALG1 was better than the performance of ALG2 if:

$$\left| f_i^{ALG1} - f_i^{ALG2} \right| < 10^{-3} \tag{17}$$

and the number of iterations (#iter), or the number of function-gradient evaluations (#fg), or the CPU time of ALG1 was less than the number of iterations, or the number of function-gradient evaluations, or the CPU time corresponding to ALG2, respectively.

In the first set of numerical experiments we compare DESCON versus DESCON-P. Figure 1 shows the Dolan and Moré [10] CPU performance profile of DESCON versus DESCON-P. In a performance profile plot, the top curve corresponds to the method that solved the most problems in a time that was within a factor τ of the best time. The percentage of the test problems for which a method is the fastest is given on the left axis of the plot. The right side of the plot gives the percentage of the test problems that were successfully solved by these algorithms, respectively. Mainly, the right side is a measure of the *robustness* of an algorithm.

When comparing DESCON versus DESCON-P subject to CPU time metric we see that DESCON-P is top performer. However, from Figure 1 subject to the number of iterations, we see that DESCON was better in 19 problems (i.e. it achieved the minimum number of iterations in 19 problems). DESCON-P was better in 4 problems and they achieved the same number of iterations in 976 problems, etc. Out of 1000 problems, only for 999 problems does the criterion (17) hold. Therefore, subject to the CPU time metric, DESCON-P appears to generate slightly the best search direction and the best steplength, on average.

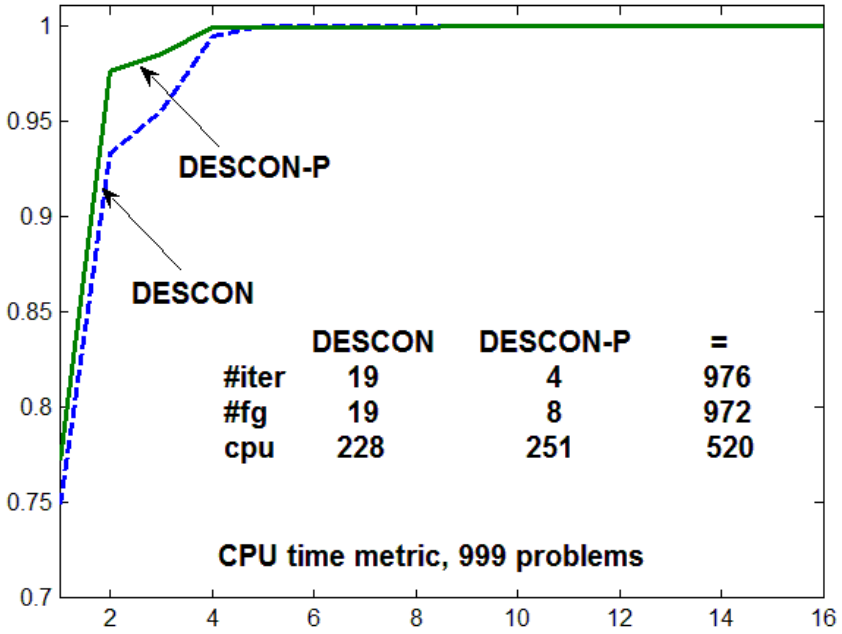

Fig. 1. Performance profies of DESCON versus DESCON-P

In the second set of numerical experiments let us compare DESCON versus DESCON-H. Figure 2 shows that subject to the CPU time metric DESCON is more efficient versus DESCON-H.

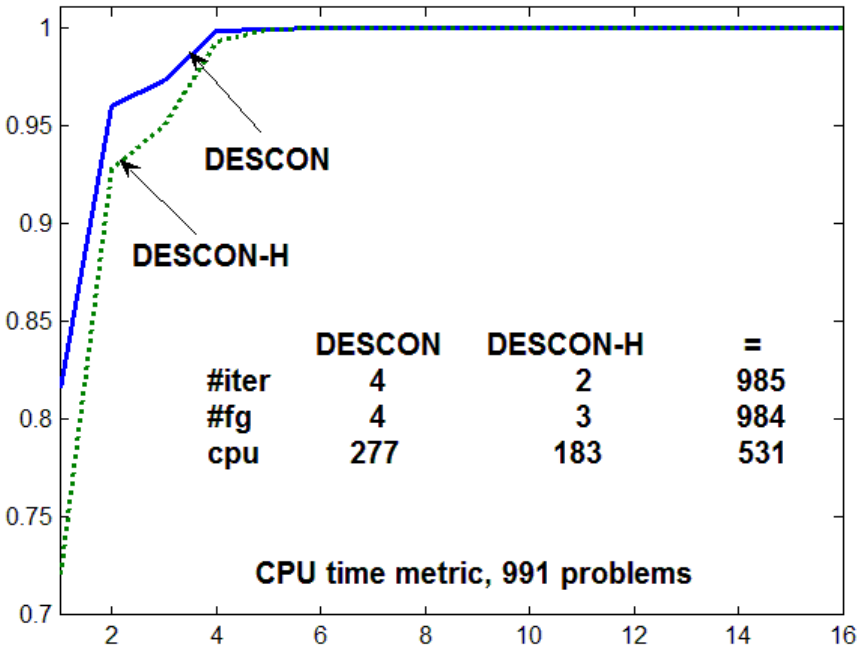

Fig. 2. Performance profiles of DESCON versus DESCON-H

In the third set of numerical experiment we compare DESCON-P versus DESCON-H. Figure 3 shows the performance profiles of these algorithms.

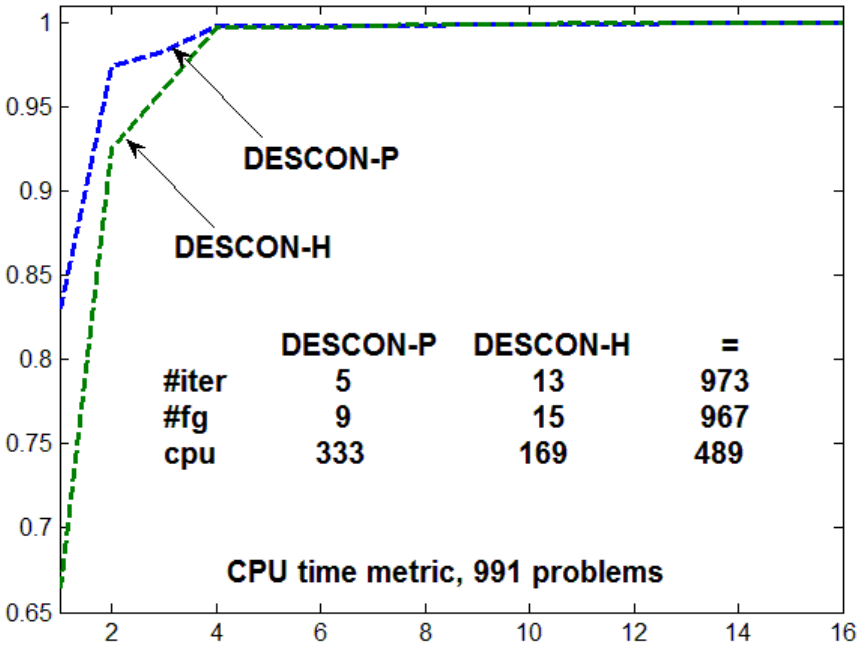

Fig. 3. Performance profiles of DESCON-P versus DESCON-H.

Figure 4 presents the global performance profiles of all these algorithms.

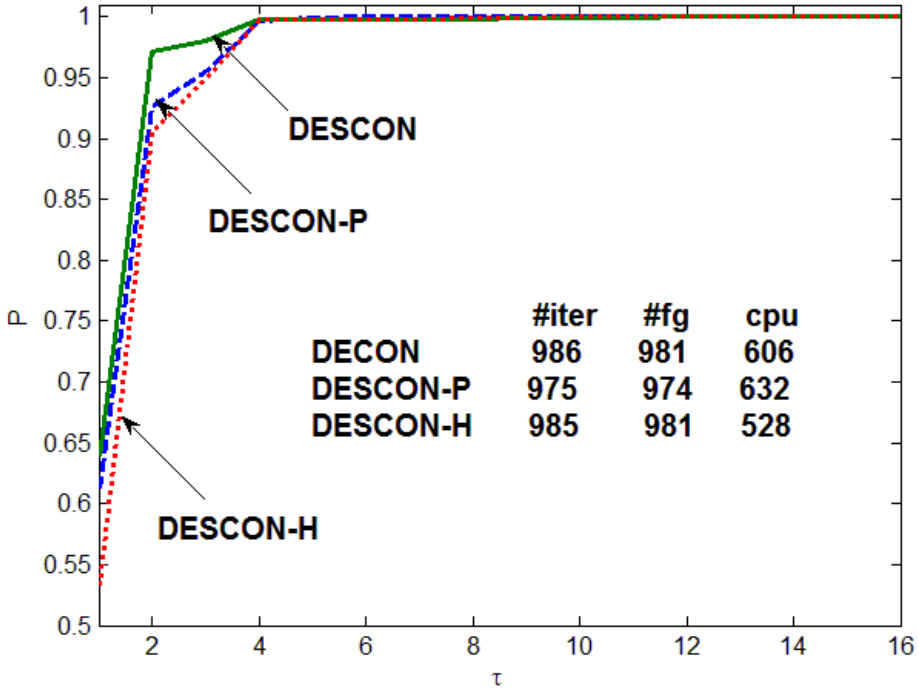

Fig. 4. Performance profiles of DESCON, DESCON-P and DESCON-H.

Conclusion

The performances of all these variants of DESCON are similar. It seems that simple modifications of the conjugate gradient parameter, we considered here, do not influence too much the efficiency and robustness of DESCON. However, the truncation of the conjugate gradient parameter like in (15) is slightly more benefic, subject to the efficiency of the algorithm.

References

1. Andrei, N., (2013). Another conjugate gradient algorithm with guaranteed descent and conjugacy conditions for large-scale unconstrained optimization. J*ournal of Optimization Theory and Applications*, vol. 159, pp. 159-182.
2. Andrei, N., (2020). Nonlinear Conjugate Gradient Methods for Unconstrained Optimization. *Springer Optimization and Its Applications (SOIA), Vol. 158*, Springer Nature, 2020.
3. Andrei, N., (2011). Another conjugate gradient algorithm with guaranteed descent and conjugacy conditions for large-scale unconstrained optimization. *Technical Report, April 11, 2011*.
4. Wolfe, P., (1969). Convergence conditions for ascent methods, *SIAM Rev*. 11 (1969) 226-235.
5. Wolfe, P., (1971). Convergence conditions for ascent methods II: some corrections, *SIAM Rev*. 13 (1971) 185-188.
6. Hager, W.W., Zhang, H., (2005). A new conjugate gradient method with guaranteed descent and an efficient line search, *SIAM Journal on Optimization*, 16 (2005) 170-192.
7. Powell, M.J.D., (1984). Nonconvex minimization calculations and the conjugate gradient methods. In D.F. Griffiths (Ed.) *Numerical Analysis* (Dundee, 1983). *Lecture Notes in Mathematics* (vol. 1066, pp. 122-141).
8. Dai, Y.H., Liao, L.Z., (2001). New conjugate conditions and related nonlinear conjugate gradient methods. *Applied Mathematics & Optimization*, vol. 43, pp. 87-101.
9. Bongartz, I., Conn, A.R., Gould, N.I.M., & Toint, Ph.L. (1995). CUTE: constrained and unconstrained testing environments. *ACM Transactions on Mathematical Software, 21*, 123-160.
10. Dolan, E.D., Moré, J.J., (2002). Benchmarking optimization software with performance profiles, *Mathematical Programming* 91, 201-213 (2002).

-----ooooo0ooooo-----