Another numerical example for CG_DESCENT conjugate gradient algorithm

Neculai Andrei

Research Institute for Informatics,
8-10, Averescu Avenue, Bucharest, Romania
E-mail: nandrei@ici.ro

In this work I present another unconstrained optimization test function for which the CG_DESCENT algorithm by Hager and Zhang [1] is unable to give a local optimal solution.
The problem is as follows:

$$\min f(x) = \sum_{i=1}^{m} \left(x_i \left(\sum_{j=1}^{n} j x_j \right) \right),$$

where $m \le n.$ The initial point is $x_0 = [1,1,...1].$ Observe that the function $f(x)$ can be written as:

$$f(x) = (x_1 + x_2 + \cdots + x_m)(x_1 + 2x_2 + \cdots + nx_n).$$

For $m = 10$ and $n = 1000, 2000, \ldots, 10000$, the CG_DESCENT with default settings gives the following results:

CG_DESCENT Algorithm. m=10.

n	iter	nfunc	ngrad	time(c)	f	gnorm	s
1000	2	55	53	0	-0.1027469705397E+82	0.1236124527131E+43	3
2000	2	55	53	0	-0.3307932962212E+83	0.9940242683663E+43	3
3000	2	55	53	0	-0.2517038156235E+84	0.3360635085083E+44	3
4000	2	55	53	1	-0.1061748181551E+85	0.7972845440600E+44	3
5000	2	55	53	2	-0.3242157193201E+85	0.1558005589961E+45	3
6000	2	55	53	0	-0.8070773819801E+85	0.2693166699822E+45	3
7000	2	55	53	1	-0.1744914964223E+86	0.4277707922519E+45	3
8000	2	55	53	2	-0.3402729725879E+86	0.6386569325247E+45	3
9000	2	55	53	2	-0.6132870824827E+86	0.9094690921549E+45	3
10000	2	55	53	3	-0.1038747651487E+87	0.1247701279967E+46	3
TOTAL	20	550	530	0.11(seconds)			

For $m = n/2$ and $n = 1000, 2000, \ldots, 10000$, the CG_DESCENT with default settings gives the following results:

CG_DESCENT Algorithm. m=n/2.

n	iter	nfunc	ngrad	time(c)	f	gnorm	s
1000	2	55	53	0	-0.5584602480522E+77	0.8723514427957E+40	3
2000	2	55	53	0	-0.4464909819682E+78	0.3488941162306E+41	3
3000	2	55	53	1	-0.1506594917682E+79	0.7849768319985E+41	3
4000	2	55	53	0	-0.3570817884807E+79	0.1395483276796E+42	3
5000	2	55	53	2	-0.6973819883908E+79	0.2180413539136E+42	3
6000	2	55	53	2	-0.1205026111040E+80	0.3139767364501E+42	3
7000	2	55	53	1	-0.1913480130747E+80	0.4273545208833E+42	3
8000	2	55	53	2	-0.2856210118694E+80	0.5581746241122E+42	3
9000	2	55	53	3	-0.4066681965306E+80	0.7064371756348E+42	3
10000	2	55	53	1	-0.5578361737946E+80	0.8721420983905E+42	3
TOTAL	20	550	530	0.12(seconds)			

For $m = n$ and $n = 1000, 2000, \ldots, 10000$, the CG_DESCENT with default settings gives the following results:

```
CG_DESCENT Algorithm. m=n.

         n     iter    nfunc    ngrad  time(c)               f                  gnorm         s
 ---------------------------------------------------------------------------------------------
      1000        2       55       53        0     -0.4818729200028E+75   0.5294622923219E+39 3
      2000        2       55       53        0     -0.3866930208930E+76   0.2122569005284E+40 3
      3000        2       55       53        2     -0.1306435569608E+77   0.4779325864167E+40 3
      4000        2       55       53        2     -0.3098333346599E+77   0.8499726963224E+40 3
      5000        2       55       53        1     -0.6053304724831E+77   0.1328377814850E+41 3
      6000        2       55       53        2     -0.1046226799162E+78   0.1913149547552E+41 3
      7000        2       55       53        1     -0.1661614190713E+78   0.2604281904626E+41 3
      8000        2       55       53        2     -0.2480584425415E+78   0.3401783968451E+41 3
      9000        2       55       53        3     -0.3532229393916E+78   0.4305648443120E+41 3
     10000        2       55       53        3     -0.4845640896250E+78   0.5315879906929E+41 3
 ---------------------------------------------------------------------------------------------
 TOTAL           20      550      530     0.16(seconds)
```

References

[1] Hager, W.W., Zhang, H., (2006) *Algorithm 851: CG_DESCENT, a conjugate gradient method with guaranteed descent*. ACM Transaction on Mathematical Software, 32(1), 113-137.

January 4, 2013