Finally, assume that the algorithm terminates at Step 4 either with $d_k = p_{i^*}$ or with $d_k = -p_{i^*}$. Then, we have

$$|g_k^T d| \geq c\|g_k\|^2,$$

and (13) follows from (11) for $c_1 = c$. Moreover, by (12), we have

$$\|d_k\| = \|p_{i^*}\| \leq \tilde{c}_2\|g_k\|,$$

and therefore we can conclude that (14) holds with $c_2 = \tilde{c}_2$. □

We can now define the following algorithm for the unconstrained minimization of f.

Truncated Newton algorithm with Nonmonotone Line Search (TNNL)

Data. x_0, $\delta > 0$, $\theta > 0$, integers $M \geq 0$ and $N \geq 1$.

Step 1. Set $k = 0$, $m(0) = 0$; compute f_0, g_0; and set $\eta_0 = \min[\theta, \|g_0\|]$.

Step 2. If $\|g_k\| \leq \delta$, stop. Otherwise, compute the Hessian matrix H_k and determine the search direction d_k by means of Algorithm TN. If $d_k = -g_k$, set $m(k) = 0$.

Step 3. Compute α_k by means of Algorithm NL. Set $x_{k+1} = x_k + \alpha_k d_k$, $k = k+1$; compute g_k; set

$$\eta_k = \min[\theta/k, \|g_k\|],$$

$$m(k) = \begin{cases} 0, & \text{if } k < N, \\ \min[m(k-1)+1, M], & \text{if } k \geq N, \end{cases}$$

and go to Step 2.

We note that $N \geq 1$ is the number of initial steps in which the standard Armijo's rule is used and $M \geq 0$ is the maximum number of previous iterates that are taken into account at each iteration during the minimization process. Obviously, for $M = 0$, the line search algorithm reduces to Armijo's method.

By assertion (c) of Theorem 2.2, the sequence $\{x_k\}$ produced by the preceding algorithm satisfies properties (a), (b), and (c) of Theorem 2.1. Moreover, by the results of Ref. 2, if $x_k \to x^*$, $H(x^*)$ is positive definite, H is Lipschitz continuous at x^*, and the parameter ε in Algorithm TN is sufficiently small, the sequence $\{x_k\}$ converges superlinearly at x^*.

3. Numerical Results

Algorithm TNNL, defined in the preceding section, has been tested on a set of standard problems, with the following choice for the parameters:

$$\delta = 10^{-5}, \quad \theta = 10^{-3}, \quad M = 10, \quad N = 1,$$

$$\sigma = 0.5, \quad \gamma = 10^{-3}, \quad \varepsilon = 10^{-8}, \quad c = 10^{-8}.$$

For each problem, the computational results are reported by specifying the number n_l of line searches required to attain convergence, the number n_f of function evaluations, and the index of computational labor

$$n_c = n_f + nn_g + mn_H,$$

where $n_g = n_l + 1$ is the number of gradient evaluations, $n_H = n_l$ is the number of Hessian evaluations, and $m \leq \frac{1}{2}n(n+1)$ is the number of equivalent function evaluations required for the computation of H. In particular, $m = 2n - 1$ for a tridiagonal Hessian matrix.

When the sequence $\{f_k\}$ produced by Algorithm TNNL with $M = 10$ is not monotonic, we add also the results obtained for $M = 0$ that correspond to the use of Armijo's line search rule in association with Algorithm TN.

Further experiments have concerned the test at Step 2 of Algorithm TN. We replaced the instructions of Step 2 with the criterion proposed in Ref. 2, that is,

$$\text{if } s_i^T H_k s_i \leq \varepsilon \|s_i\|^2, \qquad \text{set } d_k = p_i \text{ and stop.}$$

In this way, when a point x_k is encountered where H_k is not positive definite, the conjugate gradient iteration is terminated and the current estimate p_i is used. It is shown in Ref. 2 that the directions p_i produced by the algorithm satisfy an angle condition, so that we can set $d_k = p_{i+1}$ at Step 4. The algorithm that incorporates this rule will be coded as TNNLA.

For all codes, the termination criterion was $\|g_k\| \leq 10^{-5}$, although, in most cases, the last step achieves a considerably higher accuracy.

Problem 3.1. Wood Function (Ref. 18):

$$f(x) = 100(x_1^2 - x_2)^2 + (x_1 - 1)^2 + (x_3 - 1)^2 + 90(x_3^2 - x_4)^2 + 10.1[(x_2 - 1)^2 + (x_4 - 1)^2] + 19.8(x_2 - 1)(x_4 - 1),$$

$$x_0 = [-3, -1, -3, -1]^T,$$

$$x^* = [0, 0, 0, 0]^T,$$

$$f(x^*) = 0.$$

This function constitutes a difficult test for Newton's method, since a sequence of pure Newton's iterates would be attracted by a saddle point near $[-1, 1, -1, 1]^T$.

From Table 1, we note that the use of the nonmonotone line search is beneficial not only in terms of function evaluations, but also in terms of line searches. The behavior of the sequences $\{f_k\}$ produced by Algorithm TNNL for $M = 10$ and $M = 0$ is shown in Fig. 1, where the ordinate scale is logarithmic.

Table 1. Results for Problem 3.1.

Code		n_l	n_f	n_c
TNNL	$(M=10)$	27	32	414
TNNL	$(M=0)$	39	58	608
TNNLA	$(M=10)$	74	75	1115
TNNLA	$(M=0)$	81	91	1229

We observe also that Algorithm TNNL is much more effective than Algorithm TNNLA. In fact, the use of the opposite Newton direction at Step 4 of Algorithm TN prevents the sequence $\{x_k\}$ to be trapped by the saddle point near $[-1, 1, -1, 1]^T$.

Finally, we remark that the truncation criterion employed allows a complete cycle of conjugate gradient iterations to be performed at each step. A less accurate solution to the Newton equation has unfavorable effects on the performance of the algorithm. As an example, for $\theta = 10^{-1}$, we obtained the following result:

$$n_l = 34, \qquad n_f = 47, \qquad n_c = 187.$$

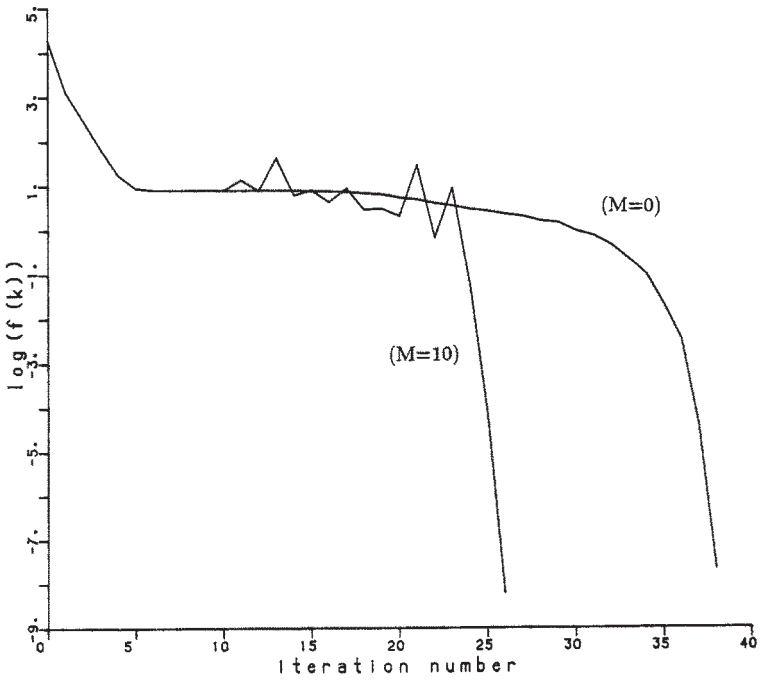

Fig. 1. Behavior of $\{f_k\}$ in Problem 3.1.

Problem 3.2. Scaled Rosenbrock Function (Ref. 18):

$$f(x) = c(x_2 - x_1^2)^2 + (1 - x_1)^2,$$

$$x_0 = [-1.2, 1]^T,$$

$$x^* = [1, 1]^T,$$

$$f(x^*) = 0.$$

For $c = 10^2$, this is a well-known test function with a steep-sided valley along the parabola $x_2 = x_1^2$. For increasing values of the parameter c, Problem 3.2 yields badly scaled variants of the Rosenbrock function.

Table 2 gives evidence for the usefulness of the nonmonotone line search technique as the ill-conditioning of the Hessian matrix increases. In Fig. 2, the behavior of $\{f_k\}$ is shown for the case $c = 10^6$.

Problem 3.3. Scaled Cube Function (Ref. 19):

$$f(x) = c(x_2 - x_1^3)^2 + (1 - x_1)^2,$$

$$x_0 = [-1.2, 1]^T,$$

$$x^* = [1, 1]^T,$$

$$f(x^*) = 0.$$

This function, which is similar to Rosenbrock's function, has a valley along the curve $x_2 = x_1^3$.

As in Problem 3.2, increasing values of the parameter c yield badly scaled variants. Table 3 shows that, also in this problem, the nonmonotone line search allows a considerable computational saving especially for large values of c.

Table 2. Results for Problem 3.2.

Code	c	n_l	n_f	n_c
TNNL $(M = 10)$	10^2	11	16	73
	10^4	11	17	74
	10^6	9	15	62
TNNL $(M = 0)$	10^2	21	29	136
	10^4	78	112	504
	10^6	350	518	2270

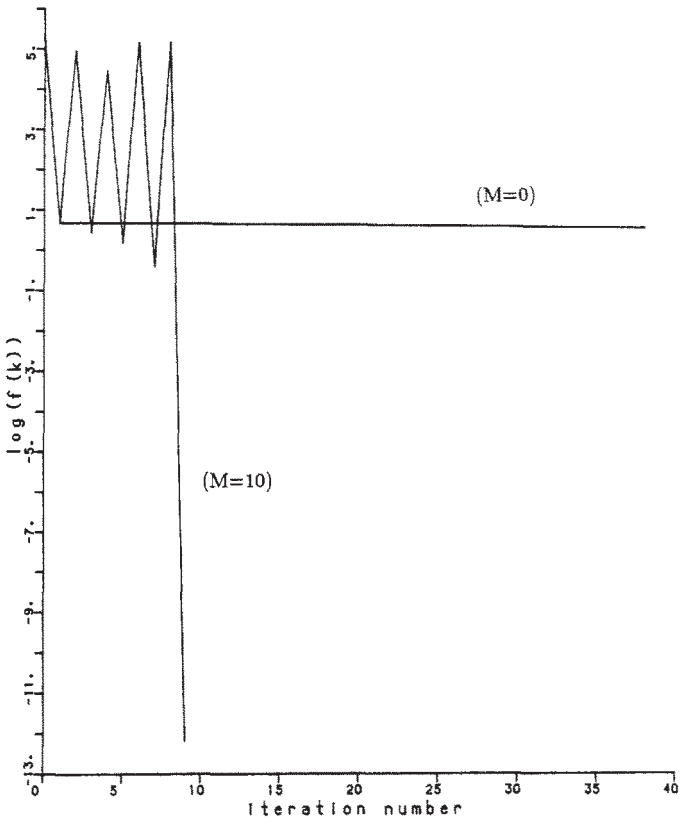

Fig. 2. Behavior of $\{f_k\}$ in Problem 3.2 for $c = 10^6$.

Table 3. Results for Problem 3.3.

Code		c	n_l	n_f	n_c
TNNL	$(M = 10)$	10^2	7	10	47
		10^4	7	10	47
		10^6	5	8	35
TNL	$(M = 0)$	10^2	26	37	169
		10^4	107	158	695
		10^6	484	722	3144

Problem 3.4. Separated Rosenbrock Function (Ref. 15):

$$f(x)=\sum_{i=1}^{n/2}[100(x_{2i}-x_{2i-1}^2)^2+(1-x_{2i-1})^2],$$

$$x_0=[-1.2, 1, \ldots, -1.2, 1]^T,$$

$$x^*=[1, \ldots, 1]^T,$$

$$f(x^*)=0.$$

This function is an n-dimensional extension of Rosenbrock's function with a block-diagonal Hessian matrix ($m=1.5n$).

From Table 4, it can be seen that, as expected, neither n_l nor n_f depend on the problem dimension.

We remark also that, in this problem, the true Newton's direction is computed by means of two conjugate gradient iterations of Algorithm TN for all values of n.

Problem 3.5. Extended Rosenbrock Function (Ref. 15):

$$f(x)=\sum_{i=1}^{n-1}[100(x_{i+1}-x_i^2)^2+(1-x_i)^2],$$

$$x_0'=[-1.2, 1, \ldots, -1.2, 1]^T,$$

$$x_0''=[2, \ldots, 2]^T,$$

$$x^*=[1, \ldots, 1]^T,$$

$$f(x^*)=0.$$

This function, which is still an n-dimensional extension of Rosenbrock's function, has a tridiagonal Hessian matrix. Two initial points, among those proposed in the literature, have been considered.

The results, obtained starting from x_0' and x_0'', are shown in Tables 5 and 6 respectively. From Table 5, we may note that the advantages of the nonmonotone line search technique are preserved also when the problem

Table 4. Results for Problem 3.4.

Code	n	n_l	n_f	n_c
TNNL ($M=10$)	2	11	16	73
	2000	11	16	73016
	20000	11	16	730016

Table 5. Results for Problem 3.5 starting from x_0'.

Code	n	n_l	n_f	n_c
TNNL ($M=10$)	10*	22	23	671
	20	42	43	2541
	100	147	148	44201
TNNL ($M=0$)	10*	27	38	831
	20	51	62	3091
	100	165	191	49626

* The termination criterion is satisfied at a point near $[-1, 1, \ldots, 1]^T$.

dimension is increased. In Table 6, we report also the maximum number $i_{\max}$ of conjugate gradient iterations required for approximating the Newton's direction with the prescribed accuracy. It is interesting to observe that, after a certain value of the problem dimension, $i_{\max}$ remains approximately constant at a value that is well beyond the value of n.

Problem 3.6. Extended Powell Function (Ref. 18):

$$f(x)=\sum_{i=1}^{n/4}[(x_{4i-3}+10x_{4i-2})^2+5(x_{4i-1}-x_{4i})^2$$
$$+(x_{4i-2}-2x_{4i-1})^4+10(x_{4i-3}-x_{4i})^4],$$
$$x_0=[3,-1,0,1,\ldots,3,-1,0,1]^T,$$
$$x^*=[0,\ldots,0]^T,$$
$$f(x^*)=0.$$

This function is an n-dimensional extension of Powell singular function and has a block-diagonal Hessian matrix ($m=2.5n$).

Table 6. Results for Problem 3.5 starting from x_0''.

Code	n	n_l	n_f	n_c	$i_{\max}$
TNNL ($M=10$)	10	11	12	341	10
	100	11	12	3401	24
	1000	10	11	31001	26
	10000	10	11	310001	26
TNNL ($M=0$)	10	20	27	617	10
	100	18	25	5507	25
	1000	14	17	43003	27
	10000	13	16	400003	29

Table 7. Results for Problem 3.6.

Code	n	n_l	n_f	n_c
TNNL ($M=10$)	4	15	16	226
	2000	18	19	128019
	20000	18	19	1280019

In this case, the full Newton step satisfies Armijo's conditions, so that $\{f_k\}$ is monotonic and the results are independent of M. Table 7 shows that, as in Problem 3.4, n_l and n_f are quite independent of the problem dimension.

Problem 3.7. Dixon Function (Ref. 16):

$$f(x)=(x_1-1)^2+\sum_{i=2}^{n} i(2x_i^2-x_{i-1})^2,$$

$$x_0=[1,\ldots,1]^T, \qquad f(x^*)=0.$$

This function has a tridiagonal Hessian matrix with a full set of distinct eigenvalues.

From Table 8, we note that the maximum number $i_{\max}$ of conjugate gradient iterations increases with n. This is due to the fact that the condition number of the Hessian matrix increases with the problem dimension. In Table 9, for $n=2000$, we report for each step the number i_k of the conjugate gradient iterations of Algorithm TN.

Table 8. Results for Problem 3.7.

Code	n	n_l	n_f	n_c	$i_{\max}$
TNNL ($M=10$)	80	7	8	1761	52
	2000	8	9	50001	421
	5000	8	9	125001	515
	10000	9	10	280001	866

Table 9. Number of conjugate gradient iterations, Problem 3.7, $n=2000$.

k	0	1	2	3	4	5	6	7	8
i_k	34	42	54	67	153	187	234	289	421

Table 10. Results for Problem 3.8.

Code	n_l	n_f	n_c
TNNL $(M=10)$	8	9	84
TNNLA $(M=10)$	14	20	149

Problem 3.8. Box Function (Ref. 18):

$$f(x)=\sum_{i=1}^{10}[e^{-t_i x_1}-e^{-t_i x_2}-x_3(e^{-t_i}-e^{-10t_i})]^2,$$

$$t_i=0.1i,$$

$$x_0=[0, 10, 20]^T,$$

$$f(x^*)=0, \quad \text{at } [1, 10, 1]^T, \quad [10, 1, -1]^T,$$

and wherever $x_1=x_2$ and $x_3=0$.

It can be observed that Box's function has an infinite set of minimizers, so that the level sets are unbounded.

Table 10 shows that Algorithm TNNL is more effective than Algorithm TNNLA.

Problem 3.9. Oren Function (Ref. 15):

$$f(x)=\left[\sum_{i=1}^{n} ix_i^2\right]^2,$$

$$x_0=[1, \ldots, 1]^T,$$

$$x^*=[0, \ldots, 0]^T,$$

$$f(x^*)=0.$$

From Table 11, it can be noted again that the truncation rule of Algorithm TN allows a considerable computational saving as the problem dimension increases.

Table 11. Results for Problem 3.9.

Code	n	n_l	n_f	n_c	$i_{\max}$
TNNL $(M=10)$	10	17	18	1133	10
	50	21	22	27897	25
	100	23	24	118574	33

Table 12. Results for Problem 3.10.

Code	n_l	n_f	n_c
TNNL ($M=10$)	5	7	32
TNNLA ($M=10$)	6	8	38

Problem 3.10. Powell (1966) Function (Ref. 4):

$$f(x)=x_1^4+x_1x_2+(1+x_2)^2,$$
$$x_0=[0,0]^T,$$
$$f(x^*)=-0.582.$$

This is the only problem, among those considered here, where the modified direction $d_k=d_{i+1}^P+d_{i+1}^N$ defined at Step 4 of Algorithm TN plays a significant role.

4. Conclusions

The numerical results reported in the preceding section compare quite favorably with those given in the literature in correspondence to various modifications of Newton's method (see, e.g., Refs. 4–6, 15, and 16). However, as remarked in Ref. 20, comparing different approaches to Newton's method is a rather difficult task, since in most of existing test problems the need for modifying Newton's method arises only a few times during the minimization process. Thus, more adequate test problems would be probably required. Nonetheless, on the basis of the authors' experience, the following indications can be given.

(i) The use of a nonmonotone line search technique may allow a considerable computational saving. In fact, in all problems where the full Newton's step does not satisfy Armijo's condition, the nonmonotone version of the algorithm outperformed the version that enforces monotonicity of $\{f_k\}$, both in terms of number of line searches and in terms of number of function evaluations. It was found that the advantages of the nonmonotone version increase especially in the solution of ill-conditioned problems.

(ii) The truncated Newton algorithm revealed itself to be a valuable tool for the solution of large dimensional problems. It would seem, however, that sufficient accuracy has to be required in the termination condition of the conjugate gradient iteration. In fact, it appears that the main advantage of the truncated Newton method lies more in the fact that a relatively small number of conjugate gradient steps is needed, in most of cases, for obtaining

a good approximation of Newton's direction, than in the possibility of accepting a poor approximation of this direction when far from the solution.

It must be remarked that, although the conjugate gradient method requires in principle n steps for obtaining the true solution, the error on the solution after a given number of steps is bounded by a term that does not depend on the problem dimension, but rather on the conditioning of the matrix to be inverted and on the number of steps performed (see, e.g., Ref. 21). On the other hand, the use of a good approximation of the Newton's direction appears to be beneficial even in the early stages of the minimization process especially in the presence of steep-sided valleys.

Another point that deserves some attention is the criterion used for ensuring that the search direction is a descent direction. It would seem that there is no need of stopping the conjugate gradient iteration if the Hessian matrix is not positive definite. A better strategy seems to be that of modifying the Newton's direction only when it does not satisfy some prescribed angle condition or the Hessian matrix is nearly singular.

(iii) In the examples that have been worked out, the algorithm proposed in this paper yields better results than those obtained by methods based on the trust region strategy, in spite of the nice features of the latter approach (see, e.g., Refs. 6 and 16). It appears that, while the trust region approach constitutes an attractive technique for modifying the Newton's direction, it may have the drawback of reducing the full Newton's step, even when the pure Newton's iteration would constitute the best strategy.

References

1. MORÉ, J. J., and SORENSEN, D. C., *Newton's Method*, Studies in Numerical Analysis, Edited by G. H. Golub, Mathematical Association of America, Washington, DC, 1984.
2. DEMBO, R. S., and STEIHAUG, T., *Truncated-Newton Algorithms for Large-Scale Unconstrained Optimization*, Mathematical Programming, Vol. 26, pp. 190–212, 1983.
3. GILL, P. E., and MURRAY, W., *Newton-Type Methods for Unconstrained and Linearly Constrained Optimization*, Mathematical Programming, Vol. 7, pp. 311–350, 1974.
4. WOLFE, M. A., *Numerical Methods for Unconstrained Optimization*, Van Nostrand-Reinhold, New York, New York, 1978.
5. FLETCHER, R., *Practical Methods of Optimization, Vol. 1*, John Wiley, New York, New York, 1980.
6. BULTEAU, J. P., and VIAL, J. P., *A Restricted Trust Region Algorithm for Unconstrained Optimization*, Journal of Optimization Theory and Applications, Vol. 47, pp. 413–435, 1985.

7. DENNIS, J. E., and SCHNABEL, R. B., *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*, Prentice-Hall, Englewood Cliffs, New Jersey, 1983.
8. SHULTZ, G. A., SCHNABEL, R. B., and BYRD, R. H., *A Family of Trust-Region-Based Algorithms for Unconstrained Minimization with Strong Global Convergence Properties*, SIAM Journal on Numerical Analysis, Vol. 22, pp. 47–67, 1985.
9. STEIHAUG, T., *The Conjugate Gradient Method and Trust Regions in Large Scale Optimization*, SIAM Journal on Numerical Analysis, Vol. 20, pp. 626–637, 1983.
10. HESTENES, M. R., *Conjugate Direction Methods in Optimization*, Springer-Verlag, New York, New York, 1980.
11. CHAMBERLAIN, R. M., POWELL, M. J. D., LEMARECHAL, C., and PEDERSEN, H. C., *The Watchdog Technique for Forcing Convergence in Algorithms for Constrained Optimization*, Mathematical Programming Study, Vol. 16, pp. 1–17, 1982.
12. GRIPPO, L., LAMPARIELLO, F., and LUCIDI, S., *A Nonmonotone Line Search Technique for Newton's Method*, SIAM Journal on Numerical Analysis, Vol. 23, pp. 707–716, 1986.
13. ARMIJO, L., *Minimization of Functions Having Lipschitz-Continuous First Partial Derivatives*, Pacific Journal of Mathematics, Vol. 16, pp. 1–3, 1966.
14. DEMBO, R. S., EISENSTAT, S. C., and STEIHAUG, T., *Inexact Newton Methods*, SIAM Journal on Numerical Analysis, Vol. 19, pp. 400–408, 1982.
15. GARG, N. K., and TAPIA, N. K., *QDN: A Variable Storage Algorithm for Unconstrained Optimization*, Department of Mathematical Sciences, Rice University, Technical Report, 1977.
16. DIXON, L. C. W., and PRICE, R. C., *Numerical Experience with the Truncated Newton Method*, Numerical Optimization Centre, Hatfield Polytechnic, Technical Report No. 169, 1986.
17. MIELE, A., and CANTRELL, J. W., *Study on a Memory Gradient Method for the Minimization of Functions*, Journal of Optimization Theory and Applications, Vol. 3, pp. 459–470, 1969.
18. MORÉ, J. J., GARBOW, B. S., and HILLSTROM, K. E., *Testing Unconstrained Optimization Software*, ACM Transactions on Mathematical Software, Vol. 7, pp. 17–41, 1981.
19. LEON, A., *A Comparison Among Eight Known Optimizing Procedures*, Recent Advances in Optimization Techniques, Edited by A. Lavi and T. Vogl, John Wiley, New York, New York, 1966.
20. MCCORMICK, G. P., *Nonlinear Programming*, John Wiley and Sons, New York, New York, 1983.
21. LUENBERGER, D. G., *Linear and Nonlinear Programming*, Addison-Wesley, Reading, Massachusetts, 1984.