Transaction on Evolutionary Algorithm and Nonlinear Optimization
ISSN: 2229-8711 Online Publication, June 2012
www.pcoglobal.com/gjto.htm
NG-O21/GJTO

Global Journal of Technology & Optimization
Volume 3, 2012

SOLVING SYSTEMS OF NONLINEAR EQUATIONS USING A GLOBALLY CONVERGENT OPTIMIZATION ALGORITHM

Sona Taheri, Musa Mammadov

Centre for Informatics and Applied Optimization, School of Science, Information Technology and Engineering, University of Ballarat, Victoria, Australia
Email: sonataheri@students.ballarat.edu.au, m.mammadov@ballarat.edu.au

Received February 2010, Revised September 2011, Accepted May 2012

Abstract
Solving systems of nonlinear equations is a relatively complicated problem for which a number of different approaches have been presented. In this paper, a new algorithm is proposed for the solutions of systems of nonlinear equations. This algorithm uses a combination of the gradient and the Newton's methods. A novel dynamic combinatory is developed to determine the contribution of the methods in the combination. Also, by using some parameters in the proposed algorithm, this contribution is adjusted. We use the gradient method due to its global convergence property, and the Newton's method to speed up the convergence rate. We consider two different combinations. In the first one, a step length is determined only along the gradient direction. The second one is finding a step length along both the gradient and the Newton's directions. The performance of the proposed algorithm in comparison to the Newton's method, the gradient method and an existing combination method is explored on several well known test problems in solving systems of nonlinear equations. The numerical results provide evidence that the proposed combination algorithm is generally more robust and efficient than other mentioned methods on some important and difficult problems.

Keywords: Systems of nonlinear equations, Newton's Method, Gradient method, Line search, Global convergence

1. Introduction

The solutions of systems of equations have a well-developed mathematical and computational theory when solving linear systems, or a single nonlinear equation. The situation is much more complicated when the equations in the system do not exhibit nice linear or polynomial properties. In this general case, both the mathematical theory and computational practices are far from complete understanding of the solution process.

Systems of nonlinear equations arise in various domains of practical importance such as engineering, medicines, chemistry, and robotics [15, 21, 37]. They appear also in many geometric computations such as intersections, minimum distance, creation of centenary curves, and when solving initial or boundary value problems in ordinary or partial differential equations [13] and [16]. The application of nonlinear systems in load flow calculation in power system has been done by Spong and et. all [32] in which their results of block Guass-Sidel iteration are compared with those of Newton-Raphson iteration. Solving such a system involves finding all the solutions of equations contained in the mentioned system.

In this paper, we consider the problem of finding solutions to a system of nonlinear equations of the form

$$F(x) = \Theta, \qquad (1)$$

where $F: R^n \to R^n, \Theta = (0, \dots, 0)$, and x refers to n variables, $x = (x_1, \dots, x_n)$. We denote the i-th component of F by f_i, where $f_i: R^n \to R$ is a nonlinear function and twice continuously differentiable on a convex set $D \subset R^n$.

There is a class of methods for the numerical solutions of the system (1), which arises from iterative procedure used for systems of linear equations [12]. These methods use reduction to simpler one-dimensional nonlinear equations for the components $f_1, f_2 \dots f_n$.

There are some iterative methods for solving systems of nonlinear equations in the book written by Kelley [15]. A wide range class of iterative methods for solving systems of nonlinear equations has been suggested in the papers [2, 11, 25, 26].

Most of the methods for solving (1) are optimization-based methods [1, 4, 6, 11, 17, 22, 37]. In the approach proposed in [22], the system (1) is transformed in to a constraint optimization problem. At each step, some equations that are satisfied at the current point are treated as constraints and the other ones as objective functions. In a strategy based on optimization methods, at each iteration, a quadratic function is minimized to determine the next feasible point to step to. The quadratic function is the squared norm of the original system.

To find a solution of (1), one can transform the system (1) into an unconstrained optimization problem and then solving the new unconstrained problem instead by applying an optimization method. The transformed problem is formulated as:

$$f(x) = \frac{1}{2}\|F(x)\|^2, \qquad (2)$$

where, here and throughout the paper, $\|\cdot\|$ stands for the Euclidean norm. Obviously, optimal solutions of problem (2) with the zero value of the objective function correspond to global solutions of system (1).

In the last decades, many publications, both in theoretical and especially numerical issues, have been done for solving the problem (2) [3, 5, 9, 10, 18, 24, 27, 31, 33, 35]. Many search

direction methods such as the gradient method, the Newton's method, the quasi-Newton methods, the conjugate gradient and coordinate direction methods have been applied to find a minimizer of (2).

The steepest descent method (or gradient method) is a commonly used method. It has the globally convergence property, however, this method suffers from the slow speed and is easy plunging into local minima. In order to accelerate these difficulties, many methods have been used [10]. One way is the use of combination of different local optimization methods. It has been found that these methods show significant reduction in the number of iterations and the expense of function evaluations. In recent years, there has been a growing interest in applying these combination methods [7, 29, 30, 36]. Buckley [7] proposed a strategy of using a conjugate gradient search direction for most iterations and using periodically a quasi-Newton step to improve the convergence. This algorithm offers the user the opportunity to specify the amount of available storage. Wang and et al. [36] proposed a revised conjugate gradient projection method, that is, a combination of the conjugate projection gradient and the quasi-Newton methods for nonlinear inequality constrained optimization problems. Recently, Y. Shi [29] proposed a combined method of the Newton's and the steepest descent methods for solving nonlinear systems of equations within each iteration. Further in [30], in order to deal with an unconstrained problem, the combination of the steepest descent with the Newton and the quasi-Newton methods were developed and compared with some traditional and existing methods.

Our procedure here for solving systems of nonlinear equations is based on the combination of local optimization methods. We apply the gradient and the Newton's methods for our combination algorithm. They are combined into an integrated procedure, and especially the dynamic combination is of our interest challenge. The combined algorithms proposed in this paper are different from the existing algorithms [7, 29, 30, 36]. In the other words, we propose a novel algorithm with a new combination which offers the user the opportunity to specify the amount contribution of the methods.

The rest of the paper is organized as follows: Section 2 gives a brief review to preliminaries about optimization. In Section 3, we review the descent methods. We present the proposed combination algorithm in Section 4. The global convergence property of this algorithm has been proved in Section 5. We have demonstrated the efficiency of the proposed algorithm with some experiments in Section 6. Section 7 concludes the paper.

2. Preliminaries

Usually, optimization methods are iterative. The basic idea is that, with an initial guess of the optimal values of the variables, $\{x_0\}$, an optimization method generates a sequence $\{x_k\}$ of improved estimates until it reaches a solution. When $\{x_k\}$ is a finite sequence, the last point is the optimal solution; when $\{x_k\}$ is infinite, it has a limit point which is the optimal solution of the problem. The strategy used to move from one iterate to the next distinguishes one algorithm from another. A typical behavior of an algorithm which is regarded as acceptable is that the iterates $\{x_k\}$ move steadily towards the neighborhood of a point local minimizer, and then rapidly converge to that point. When a given convergence rule is satisfied, the iteration will be terminated. In general, the most natural stopping criterion is

$$\|g_k\| < \epsilon, \quad (3)$$

where g_k stands for $\nabla f(x)$ at x_k and f is defined by (2). $\epsilon > 0$ is a prescribed error tolerance.

Let x_k be the k-th iterate, d_k k-th search direction, and α_k k-th step length, then the k-th iteration is

$$x_{k+1} = x_k + \alpha_k d_k. \quad (4)$$

There are two fundamental strategies for moving from the current point x_k to a new state x_{k+1}: Trust region [24, 38] and Line search [24, 28, 31, 33]**.**

In the trust region strategy, the information gathered about f is used to construct a model function whose behavior near the current point x_k is similar to that of the actual objective function f. When x is far from x_k, the model may not be a good approximation of f. Therefore, the search for a minimizer of the model is restricted to some region around x_k.

In the line search strategy, the algorithm chooses a direction d_k and searches along this direction from the current iterate x_k for a new iterate with a lower function value.

The line search and trust-region approaches differ in the order in which they choose the direction and distance of the move to the next iterate. Line search starts by fixing the direction d_k and then identifying an appropriate distance, namely the step length α_k. In trust region, firstly a maximum distance is chosen, the trust region radius, and then a direction and a step that attain the best possible improvement subject to this distance constraint is found. If this step proves to be unsatisfactory, the distance measure will be reduced and tried again [24].

A trust region method is effective since it limits the step to a region of greater confidence in the local model and attempts to utilize more information from the local model for finding a shortened step. However, trust region models are more difficult to formulate and solve than a line search strategy [31]. In this paper, we will focus on line search strategies.

2.1 Line Search

Line search methods are traditional and efficient methods for solving unconstrained minimization problems. Its convergence has attracted more attention in recent years [3, 19, 35].

The success of a line search method depends on effective choices of both the direction d_k and the step length α_k. It is clarified that the search direction plays a main role in the algorithm and that step length guarantees the global convergence in some cases.

There are two alternatives for finding the distance to move along d_k namely the exact line search and inexact line search [19, 28, 31, 33]. In the exact line search, the following one-dimensional minimization problem will be solved to find a step length α*:*

$$min_\alpha f(x_k + \alpha d_k). \quad (5)$$

If we choose α_k such that the objective function has acceptable descent amount, i.e., it means the descent $f(x_k) - f(x_k + \alpha_k d_k) > 0$ (6)

is acceptable by users, such a line search is called inexact line search. Since, in practical computation, exact optimal step length generally cannot be found, and it is also expensive to find almost exact step length, therefore the inexact line search with less computation load is highly popular.

A simple condition we could impose on α_k in an inexact line search is to require a reduction in f:

$$f(x_k + \alpha_k d_k) < f(x_k). \quad (7)$$

It has been shown that this requirement is not enough to produce convergence to optimal point [24, 33]. The difficulty is that there is not always a sufficient reduction in f at each step, a concept we discuss next.

There are several inexact line search rules for choosing an appropriate step length α_k, for example the Armijo rule, the

Goldstein rule, and the Wolfe-Powell rules [24, 31, 33], which are described briefly in the following.

Armijo Rule and Goldstein Rule

Armijo rule is as follows:

$f(x_k) - f(x_k + \beta^m \tau d_k) \geq -\rho\beta^m \gamma g_k^T d_k,$

where $\beta \in (0,1)$, $\rho \in \left(0, \frac{1}{2}\right)$, and $\gamma > 0$, (8)

$m = 0,1, \ldots$, are tried successively until the above inequality is satisfied for $m = m_k$.

Goldstein presented the following rule. Let

$$I = \{\alpha > 0 : f(x_k + \alpha d_k) < f(x_k)\}$$

be an interval. In order to guarantee the function decreases sufficiently, we want to choose α such that it is away from the two end points of the interval I.

The Goldstein conditions are

$f(x_k + \alpha d_k) \leq f(x_k) + \rho\alpha g_k^T d_k,$ (9)

and

$f(x_k + \alpha d_k) \geq f(x_k) + (1-\rho)\alpha g_k^T d_k,$ (10)

which exclude those points near the right end point and the left end point.

Wolfe-Powell Rule

It is possible that the rule (10) excludes the minimizing value of α outside the acceptable interval. Instead, the Wolfe-Powell gives another rule to replace (10):

$$g_{k+1}^T d_k \geq \sigma g_k^T d_k, \sigma \in (\rho, 1).$$

Therefore, the step length α_k in the Wolfe-Powell rule will be determined along the direction d_k satisfying:

$f(x_k + \alpha d_k) \leq f(x_k) + \rho\alpha g_k^T d_k,$ (11)

and

$g_{k+1}^T d_k \geq \sigma g_k^T d_k, \sigma \in (\rho, 1).$ (12)

The Wolfe-Powell rule is a popular inexact line search rule. We will use it in our algorithm and all experiments in this paper.

2.2. Search Directions

The search direction in gradient-based methods often has the form

$d_k = -B_k^{-1} g_k,$ (13)

where B_k is a symmetric and nonsingular matrix. For example, in the gradient method B_k is simply the identity matrix, $d_k = -g_k$ [3, 24, 33]. $d_k = -H_k^{-1}\mathrm{g}_k$ corresponds to the Newton's method with H_k^{-1} being available, where H_k is an exact Hessian of f [24, 33]. In quasi-Newton methods, B_k is an approximation to the Hessian H_k that is updated at every iteration by means of a low-rank formula [5, 9, 24, 33]. In the conjugate gradient method, d_k is defined by

$d_k = -g_k + \beta_k d_{k-1}$, $k \geq 2$, and $d_1 = -g_1$; β_k is a parameter [8,18, 24, 33].

When d_k is defined by (13) and B_k is positive definite, we have $d_k^T g_k = -g_k^T B_k^{-1} g_k < 0$, and therefore d_k is a descent direction.

The search direction d_k is generally required to satisfy the descent condition:

$g_k d_k < 0.$ (14)

The condition (14) guarantees that d_k is a descent direction of f at x_k [24, 33].

3. Descent Methods

Many techniques have been devoted for solving (2), as well as (1). These problems are usually carried out using iterative methods due to the fact that there are generally no analytical methods to solve these problems. Among the variety of the exiting methods, the descent direction methods are the most popular techniques because of their fast convergence property. A general descent direction algorithm is given in the Algorithm 1.

Algorithm 1. A General Descent Framework

0. Lets $x_1 \in R^n$ be a given initial point, and $\epsilon > 0$ an error tolerance. Each iteration $k = 1,2, \ldots$ of a descent direction method contains the following steps:
1. If $\|g_k\| < \epsilon$, then stop.
2. Compute a descent direction d_k at x_k satisfying (14).
3. Determine an appropriate step length $\alpha_k > 0$.
4. Set $x_{k+1} = x_k + \alpha_k d_k$, and go to the next iteration.

Let $\Omega = \{x | f(x) \leq f(x_1)\}$ be the level set, and consider the Wolfe-Powell conditions (11) and (12) to determine α_k, then the global convergence of the Algorithm 1 is given by the following Theorem [33].

Theorem 1. Let α_k in the above descent direction algorithm be defined by (11) and (12). Let also d_k satisfies

$$\cos\theta_k \geq \delta,$$ (15)

for some $\delta > 0$ and for all k, where θ_k is the angle between d_k and $-g_k$. If $g(x)$ exists and is uniformly continuous on the level set Ω, then either $g_k = 0$ for some k, or $f_k \to -\infty$, or $g_k \to 0$.

Proof can be found in [33], Theorem 2.5.4.

One of the most widely used methods satisfying Theorem 1 is the gradient method, in which $\mathrm{d}_k = -g_k$ for all k. Although the method is globally convergent and usually works well in some early steps, as a stationary point is approached, it may descend very slowly. In fact, it is shown that the convergence rate of the gradient method is at least linear, and the following bound holds

$\frac{\|x_{k+1} - x^*\|}{\|x_k - x^*\|} \leq \frac{\lambda_{max} - \lambda_{min}}{\lambda_{max} + \lambda_{min}}$, (16)

where λ_{max} and λ_{min} are the largest and the smallest eigenvalues of the Hessian matrix, respectively.

In order to cope with the above-mentioned difficulties, one can use the Newton's method with the quadratic convergence property. At the k-th iteration, the classical Newton's direction is the solution of the following system:

$$H_k d_k = -g_k,$$ (17)

where H_k is the Hessian matrix at x_k. If H is positive definite, then the Newton's direction is a descent direction and consequently the system has a unique solution. Even when H is positive definite, it is not guaranteed that Newton's method will be globally convergent. Although the Newton's method generally converges faster than the gradient method, it depends strongly on a starting point. On the other hand, the application of the Newton's method for solving the nonlinear equations is expensive due to the direct calculations of second order derivatives of the function, H. A number of techniques avoiding the direct computation of H may be used. Upon different approximation there are different methods. In this category are the quasi-Newton methods which approximate second derivatives in a most subtle and efficient way. Another alternative is the use of a fusion of different local optimization methods which lead naturally to powerful algorithms and has been attracted extensive attention in recent years. One of the most successful methods of this category, introduced by Shi [30], uses a combination of the gradient method and the Newton's method. This algorithm is an efficient algorithm for solving problem (2) due to its global convergence property. In our experiments, we compare our results with this combination algorithm and refer it by ShA. The direction in algorithm ShA is very close to the Newton's direction. However, practical

implementations show that, in some cases the gradient method can be a more suitable choice than the Newton's method. For instance, when the difference of the function values, in two previous iterations, and also the value of the gradient in the previous iteration is large enough, the gradient method may work better than the Newton's method.

4. Proposed Algorithm

Our aim here is to present an algorithm with two different combinations for solving the problem (2), as well as the problem (1). Both proposed combinations are constructed so that they satisfy in the condition of descent methods and as well as the Theorem 1.

Let δ_0, η, ρ and σ be four parameters so that $0 < \delta < 1$, $0 < \eta < 1$, $0 < \rho < \frac{1}{2}$ and $\rho < \sigma < 1$. Take any positive constants γ_1, γ_2 and b_i, $i = 1,2,3$, such that $\gamma_i > 1$, $0 < b_1 < 1$, $1 < b_2 < 1/\delta$ and $b_3 > 1$ and initialize Λ_0 by 1. Let also T, and τ be very large and small positive numbers, respectively, and let $f(x_0) = f(x_1)$. The steps of the proposed algorithm are as follows.

Algorithm 2. A Combination of the Gradient and Newton's Methods

0. Choose a starting point $x_1 \in R^n$, and an error tolerance $\epsilon > 0$. For $k = 1,2, \ldots$ do
1. If $\|g(x_k)\| < \epsilon$, then stop.
2. If the Newton's direction d_1 is not computable, due to the singularity of the Hessian, then compute the gradient direction $d_2 = -g_k$ at x_k, and go to step 8.
3. Compute the gradient direction d_2 and the Newton's direction d_1 at x_k that satisfies (17).
4. Set $\delta = \delta_0$ and $\Lambda = \Lambda_0$. If $|f(x_k) - f(x_{k-1})| > \gamma_1$ and $g_k > \gamma_2$, set $\delta \leftarrow b_2\delta_0$ and go to step 7.
5. If $k = 1$ or if $\|g_k\| \le \|g_{k-1}\|$, set $\overline{x} = x_k + d_1$ and go to step 6.
6. If $f(\overline{x}) < f(x_k)$ and $g(\overline{x}) \le \eta g_k$ then $\delta \leftarrow b_1\delta_0$.
7. If $d_1^T d_2 \ge 0$ go to step 9, otherwise go to the next step.
8. Use rules (11) and (12) to determine a step length $\alpha_k > 0$ along the direction $d_k = d_2$, set $s_k = \alpha_k d_k$ and go to step 12.
9. Compute ξ as follows:
$$\xi = \frac{1}{\Lambda + |f(x_k) - f(x_{k-1})|} \qquad (18)$$
and set $d(\xi) = (1 - \xi)d_2 + \xi d_1$.
10. If $d(\xi)^T d_2 < \delta\|d(\xi)\|\|d_2\|$, set $\Lambda \leftarrow b_3\Lambda$ and and go back to step 9.
11. Consider one of the following two versions to calculate s_k:
    a. Use rules (11) and (12) to determine a step length $\alpha_k > 0$ along the direction $d_k = d_2$ and set $\widetilde{s_k} = \alpha_k(1 - \xi)d_2 + \xi d_1$. If $f(x_k + \widetilde{s_k}) \le f(x_k) - \tau\|\widetilde{s_k}\|$ and $\alpha_k\|d_2\| \le T\|d_1\|$, set $s_k = \widetilde{s_k}$; otherwise set $s_k = \alpha_k d_2$.
    b. Use rules (11) and (12) to determine a step length $\alpha_k > 0$ along the direction $d_k = d(\xi)$ and set $s_k = \alpha_k d_k$.
12. Set $x_{k+1} = x_k + s_k$.

Parameters b_1, b_2 and b_3 are positive constants so that they offer the user the opportunity to specify the amount contribution of the methods. More precisely, when the slope of the function is slight, the algorithm tends to the Newton's method, otherwise the contribution of gradient is increased and is considered close to the gradient method. In (18), when a difference between two previous values of the function is high then ξ is close to 0, and $\xi \to 1$ as $|f(x_k) - f(x_k)| \to 0$. Moreover, this equation is a dynamic form and has a crucial rule in the algorithm so that it specifies the amount contribution of the methods. It, also, guaranties that, near the solution, we get the optimal point with a super-linear convergence rate.

In step 11 of the above algorithm, we use two different strategies by means of the combination. Step $11.a$ is a new combination and different from the existing methods in the literature. In this combination, the step length α_k is determined only along the gradient direction. In other words, we use a novel combination of the pure Newton's method (i.e., $\alpha_k = 1$) and the gradient method. The second one is the usual combination which has been developed in some research works. The step length in this case is found along a combination of the gradient and the Newton's directions.

5. Global convergence Theorem

Here, we establish the global convergence of the proposed combination algorithm based on the global convergence property of the Theorem 1.

Theorem 2. Consider using the Algorithm 2 to solve the problem (2). Assume that $g(x)$ exists and is uniformly continuous on the level set Ω. Then either $g_k = 0$ for some k, or $f_k \to -\infty$, or $g_k \to 0$.

Proof. Let assume that $g_k \neq 0$ and f_k is bounded below for all k. It is clear that in this case, $f_k < f_{k-1}$ for all k. Denote $\delta^* \leftarrow b_1\delta_0$, clearly $\delta^* \in (0,1)$. We will show that the direction d_k obtained by the algorithm satisfies condition (15) of the Theorem 1 for δ^*; that is,

$$cos\,\theta_k \ge \delta^*, \qquad (19)$$

for all k, where θ_k is the angle between d_k and $d_2 = -g_k$.

Suppose s_k is obtained at Step 8. Then $d_k = -g_k$ (Step 8), and it is easy to see that $cos\,\theta_k = 1 > \delta^*$; it means (15) holds. Now, we consider other cases: case 1: s_k is obtained via Step 11.b and case 2: s_k is obtained via Step 11.a. We will proof each case separately as follows:

Take any integer k.

1. In this case, we assume that s_k is obtained at Step 11.b. Then $d_k = d(\xi)$ is chosen as a descent direction and according to steps 9-10, the number ξ can be chosen so that the inequality $d(\xi)^T d_2 \ge \delta\|d(\xi)\|\|d_2\|$ holds, where $\delta \leftarrow b_1\delta_0$ or $\delta \leftarrow b_2\delta_0$. Therefore, we have
$$cos\,\theta_k \ge \frac{d(\xi)^T d_2}{\|d(\xi)\|\|d_2\|} \ge \delta \ge \delta^*, \qquad (20)$$
that is, (19) holds and therefore the obtained direction, d_k, satisfies in the assumption of the Theorem 1, hence the remainder proof is similar to the proof of the Theorem 1 in [33].

2. In this case, we assume s_k is obtained at Step 11.a, i.e. $s_k = \alpha_k(1 - \xi)d_2 + \xi d_1$.

If the number of cases in s_k obtained by Step 11.a is finite, then it means s_k is defined by the gradient direction, d_2, for all sufficiently large k, and therefore the proof will be easily obtained.

Now suppose it is not finite, i.e., there is a subsequence $k_m \to \infty$ such that s_{k_m} is obtained via Step 11.a.

By considering the first condition in 11.a, since f_k is bounded below we have
$$\|s_{k_m}\| \to 0 \text{ as } k_m \to \infty.$$
In addition, from $d_1 d_2 \ge 0$ we obtain
$$\|s_{k_m}\|^2 \ge \alpha_{k_m}^2(1 - \xi)^2\|d_2\|^2 + \xi^2\|d_1\|^2.$$
Now, we are going to show $g_k \to 0$. Suppose it is not true. Then there exist a subsequence $\{k_m\}$ such that $\|g_{k_m}\| \ge \bar{\epsilon} > 0, \forall k_m$.

Here we consider two cases: (i) $\|d_1\| \to 0$ as $k_m \to \infty$, and (ii) $\|d_1\|$ does not converge to zero. The case (i) leads to contradiction by applying the second condition in Step 11.a. In the case (ii), let us consider $\|d_{1,k_m}\| \geq \tilde{\epsilon} > 0, \forall k_m$ which is contradiction by $\|s_{k_m}\| \to 0$. Therefore, the proof is complete, i.e., $g_k \to 0$

6. Experiments and Results

We have evaluated the performance of the proposed algorithm for several well known benchmark test problems given in [20, 34].In the proposed algorithm, we use two different combination as described in steps 11.a and 11.b; we refer these cases as 'Ala' and 'Alb', respectively. The group of methods we have compared includes Ala, Alb, the gradient method (GM), the Newton's method (NM), and ShA presented in [8]. In all algorithms we use the Wolfe-Powell line search rules to find an acceptable step length.

The calculations were carried out using MATLAB. The comparison of the methods is based on the following criteria: all methods are terminated if the gradient converges to a predefined tolerance, $\|g(x_k)\| < \epsilon$, $\epsilon = 10^{-6}$, or the iteration number exceeds 500.

The parameters δ_0, Λ_0, η, ρ, σ, γ_i, b_i, τ and T used in this paper are: $\delta_0 = 0.001$, $\Lambda_0 = 1$, $\eta = 0.99$, $\rho = 0.001$, $\sigma = 0.9$, $\gamma_1 = \gamma_2 = n$, $b_1 = 0.01$, $b_2 = \frac{1}{b_1}$, $b_3 = 1.1$, $\tau = 10^{-10}$and $T = 10^{10}$.

We have listed the following ten test problems used in the experiments. To define the test functions, the general formats 1 to 3 have been adopted [20, 34]:

1. Dimension, n .
2. Function definition, $f_1, f_2 \ldots f_m$.
3. Standard initial point, x_0.

Problem 1. Helical Valley function

$n = m = 3$

$f_1(x) = 10[x_3 - 10\theta(x_1, x_2)]$

$f_2(x) = 10[(x_1^2 +, x_2^2)^{0.5} - 1]$

$f_3(x) = x_3$

where

$$\theta(x_1, x_2) = \begin{cases} \frac{1}{2\pi}\arctan\left(\frac{x_2}{x_1}\right), & if\ x_1 > 0 \\ \frac{1}{2\pi}\arctan\left(\frac{x_2}{x_1}\right) + 0.5, & if\ x_1 < 0 \end{cases}$$

$x_0 = (-1,0,0).$

Problem 2. Powell Singular function

$n = m = 4$

$f_1(x) = x_1 + 10x_2$

$f_2(x) = 5^{0.5}(x_3 - x_4)$

$f_3(x) = (x_2 - 2x_3)^2$

$f_4(x) = 10^{0.5}(x_1 - x_4)^2$

$x_0 = (3, -1,0,1).$

Problem 3. Wood function

$n = 4, \quad m = 6$

$f_1(x) = 10(x_2 - x_1^2)$

$f_2(x) = 1 - x_1$

$f_3(x) = 90^{0.5}(x_4 - x_3^2)$

$f_4(x) = 1 - x_3$

$f_5(x) = 10^{0.5}(x_2 + x_4 - 2)$

$f_6(x) = 10^{-0.5}(x_2 - x_4)$

$x_0 = (-3, -1, -3, -1).$

Problem 4. Watson function

$2 \leq n \leq 31, \quad m = 31$

$$f_i(x) = \sum_{j=2}^{n}(j-1)\,x_j\,t_i^{j-2} - \left(\sum_{j=1}^{n} x_j\,t_i^{j-1}\right)^2 - 1$$

$$t_i = \frac{i}{29}, \quad 1 \leq i \leq 29$$

$f_{30}(x) = x_1,$

$f_{31}(x) = x_2 - x_1^2 - 1$

Problem 5. Extended Kearfott function

$n = m = 7$

$f_i(x) = x_i^2 - x_{i+1}$

$f_n(x) = x_n^2 - x_1$

$x_0 = (0.1, 0.1, \ldots, 0.1).$

Problem 6. Extended Eiger-Sikorski-Stenger

$n = m = 9$

$f_i(x) = (x_i - 0.1)^2 + x_{i+1} - 0.1$

$f_n(x) = (x_n - 0.1)^2 + x_1 - 0.1$

$x_0 = (-2000, \ldots, -2000).$

Problem 7. Variably dimensional function

n variable, $\quad m = n + 2$

$f_i(x) = x_i - 1, i = 1, \ldots, n$

$$f_{n+1}(x) = \sum_{j=1}^{n} j(x_j - 1)$$

$f_{n+2}(x) = \left(\sum_{j=1}^{n} j(x_j - 1)\right)^2$

$x_0 = \gamma_j,$ where $\gamma_j = 1 - \left(\frac{j}{n}\right).$

Problem 8. Discrete Boundary Value function

n variable, $\quad m = n$

$$f_i(x) = 2x_i - x_{i-1} - x_{i+1} + \frac{h^2(x_i + t_i + 1)^3}{2}$$

where $h = \frac{1}{n+1}, t_i = ih, and\ x_0 = x_{n+1} = 0$

$x_0 = \gamma_j$ where $\gamma_j = t_j(t_j - 1).$

Problem 9. Extended Rosenbrock function

n variable but even, $\quad m = n$

$f_{2i-1}(x) = 10(x_{2i} - x_{2i-1}^2)$

$f_{2i}(x) = 1 - x_{2i-1}$

$x_0 = \gamma_j$ where $\gamma_{2j-1} = -1.2, \gamma_{2j} = 1.$

Problem 10. Trigonometric function

n variable, $\quad m = n$

$$f_i(x) = n - \sum_{j=1}^{n} cos\,x_j + i(1 - cos x_i) - sin x_i$$

$x_0 = \left(\frac{1}{n}, \frac{1}{n}, \ldots, \frac{1}{n}\right).$

Table 1 lists the performance of the above-mentioned algorithms relative to the number of iterations used. We have multiplied the given initial points by 10 to have an additional initial point. In this table, "TP" and "IP" stand for test problem and initial point, respectively. Table 2 shows the summary of convergence results for the Table 1. In order to compare the algorithms with more initial points, we have generated 50 random initial points uniformly distributed from their domains with the intersection of $[-10,10]$. The summary of the convergence results of the algorithms considering these random

initial points is given in Table 3. In these tables, notations "AC" and "NC" stand for the almost convergence and not convergence, respectively. Convergence means that the method finds the solution and $\|g(x_k)\| < 10^{-6}$, almost convergence means that the method finds a solution almost close to the optimal local solution and $10^{-6} < \|g(x_k)\| \leq 10^{-2}$; otherwise not convergence.

Table 1. Number of iterations for 10 test problems

TP	n	IP	Ala	Alb	ShA	NM	GM
P1	3	x_0	11	11	13	25	AC
		$10x_0$	13	21	21	NC	NC
P2	4	x_0	7	10	11	11	135
		$10x_0$	9	15	17	19	AC
P3	4	x_0	20	AC	AC	NC	NC
		$10x_0$	19	AC	NC	NC	NC
P4	6	x_0	10	12	12	75	AC
		$10x_0$	11	18	21	77	AC
P5	7	x_0	13	10	9	NC	10
		$10x_0$	8	11	15	NC	13
P6	10	x_0	10	5	7	NC	6
		$10x_0$	17	11	14	NC	8
P7	10	x_0	14	17	23	24	56
		$10x_0$	27	34	37	38	AC
P8	20	x_0	4	5	3	4	AC
		$10x_0$	12	14	7	8	NC
P9	100	x_0	15	17	26	NC	AC
		$10x_0$	57	73	78	NC	NC
P10	100	x_0	19	21	22	AC	129
		$10x_0$	23	24	27	34	472

Te numerical results in Tables 1 to 3, demonstrate the high performance of the proposed combination algorithm compared to other mentioned methods. This is confirmed by the number of iterations obtained, and the convergence properties. For example, the proposed algorithm, Ala, converges in all test problems for two different initial points. Alb converges in nine test problems out of ten. This algorithm finds the solution in the Wood function almost near the optimal solution. Although the algorithm proposed by Shi, ShA, convergences in nine test problems out of ten, but it fails to find the solution in the problem 3. Also, the number of iterations obtained by ShA is more than the proposed algorithms, in average. This is worse for the Newton's and the gradient methods with more AC and NC properties.

Table 2. Summary of convergence results for Table 1

Algorithm	Convergence	AC	NC
Ala	100.00	0.00	0.00
Alb	90.00	10.00	0.00
ShA	90.00	5.00	5.00
NM	50.00	5.00	45.00
GM	40.00	35.00	25.00

Table 3. Convergence results, by considering 50 initial random points for each test problem

Algorithm	Convergence	AC	NC
Ala	96.40	2.80	0.80
Alb	88.80	9.40	1.80
ShA	87.80	4.40	7.80
NM	54.60	2.20	43.20
GM	51.70	30.40	17.90

7. Conclusion

A combined algorithm of the gradient and the Newton's methods has been presented for solving systems of nonlinear equations. We have considered two different combinations. One of them is a usual case which has been recently introduced in some research works. Another one is a new combination and different from others in the literature. According to the numerical experiments, it is clear the proposed algorithm, especially the proposed algorithm with the new combination, is more efficient than others.

References

[1] S. Abbasbandy, Y. Tan, and S.J. Liao, "Newton-homotopy analysis method for nonlinear equations", Appl. Math. Comput. 188, 2007, pp. 1794–1800.

[2] F. Awawdeh, "On new iterative method for solving systems of nonlinear equations", Springer Science, Business Media, LLC 2009.

[3] D.P. Bertsekas, J.N. Tsitsiklis, "Gradient convergence in gradient methods with errors", SIAM J. Optim. 10, 2000, pp. 627–642.

[4] J. Biazar, and B. Ghanbari, "A modification on Newtons method for solving systems of nonlinear equations", World Academy of Science, Engineering and Technology, Vol. 58, 2009, pp. 897-901.

[5] C. G. Broyden, "Quasi-Newton methods and their application to function minimization", Math. Comput. 21, 1967, pp. 368–381.

[6] A. G. Buckley, "Modified Quasi-Newton methods for solving systems of linear Equations", Int. J. Contemp. Math. Sci., Vol. 2, no. 15, 2007, pp. 737-744.

[7] A. G. Buckley, "A combined conjugate-gradient quasi-Newton minimization algorithm", Mathematical Programming, 15, 1978, pp. 200-210.

[8] Y.H. Dai, J. Han, G. Liu, D. Sun, H. Yin, Y.-X. Yuan, "Convergence properties of nonlinear conjugate gradient methods", SIAM J. Optim. 10, 1999, pp. 345–358.

[9] J. E. Dennis, and J. More, "Quasi-Newton methods: motivation and theory", SIAM Rev. 19, 1977, pp. 46–84.

[10] R. Fletcher, 'Practical methods of optimization" , Second Edition, Wiley, New York, 1987.

[11] A. Golbabai, and M. Javidi, "Newton-like iterative methods for solving system of non-linear equations", Appl. Math. Comput. 192, 2007, pp. 546–551.

[12] C. Grosan, and A. Abraham, "A new approach for solving non linear equations systems", IEEE transactions on systems, Vol. 38, No. 3, 2008

[13] H. K. Gummel, "A self-consistent iterative scheme for one-dimensional steady state transistor calculations", IEEE Trans. Electron Devices ED-11, 1964, pp. 455-465.

[14] D. Kaya, and S. M. El-Sayed, "Adomian's decomposition method applied to systems of nonlinear algebraic equations". Appl. Math. Comput. 154, 2004, pp. 487–493.

[15] C. T. Kelley, “Iterative methods for linear and nonlinear equations”, Society for Industrial and Applied Mathematics, Philadelphia, 1995.
[16] T. Kerkhoven, “A proof of convergence of Gummel’s algorithm for realistic boundary condition”, SIAM J. Numer. Anal., 23(6), 1986, pp. 1121-1137.
[17] J. Kou, “A third-order modification of Newton method for systems of nonlinear equations”, Appl. Math. Comput, Vol. 191, 2007, pp. 117-121.
[18] G. H. Liu, L.L. Jing, L.X. Han, D. Han, “A class of nonmonotone conjuagte gradient methods for unconstrained optimization”, J. Optim. 101, 1999, pp. 127–140.
[19] J. M. Moguerza, and F.J. Prieto, “Combining search directions using gradient flows”, Math Program. 96, 2003, pp. 529–559.
[20] J. J. More, B.S. Garbow, and K.E. Hillstrom, “Testing unconstrained optimization software”, ACM Transactions on Mathematical Software, 7, 1981, pp. 17-41.
[21] G. H. Nedzhibov, “A family of multi-point iterative methods for solving systems of nonlinear equations”, Comput. Appl. Math. 222, 2008, pp. 244–250.
[22] P. Y. Nie, “A null space methods for solving system of equations”, Appl. Math. Comput., Vol. 149, No. 1, 2004, pp. 215-226.
[23] J. Nocedal, “Theory of algorithms for unconstrained optimization”, Acta Numer. 1, 1992, pp. 199–242.
[24] J. Nocedal, and J. W. Stephen, “Numerical optimization”, Springer-Verlag, New York, 1999
[25] M. A. Noor, and K. I. Noor, “Iterative schemes for solving nonlinear equations”, Appl. Math. Comput. 183, 2006, pp. 774–779.
[26] M. A. Noor, “New classes of iterative methods for nonlinear equations”, Applied Mathematics and Computation 191, 2007, pp. 128–131
[27] F. A. Potra, and Y. Shi, “Efficient line search algorithm for unconstrained optimization”, J. Optim. 85,1995, pp. 677–704.
[28] M. J. D. Powell, “Direct search algorithms for optimization calculations”, Acta Numer. 7, 1998, pp. 287–336.
[29] Y. Shi, “A globalization procedure for solving nonlinear systems of equations”, Numerical Algorithms, 12, 1996, pp. 273-286
[30] Y. Shi, “Globally convergent algorithms for unconstrained optimization”, Computational Optimization and Applications, 16, 2000, pp. 295-308
[31] Z. J. Shi, “Convergence of line search methods for unconstrained optimization”, Applied Mathematics and Computation 157, 2004, pp. 393–405.
[32] M. I. Spong, I. Norman Katz, H. Dai, and J. Zaborszky, “Block diagonal dominance for systems of nonlinear equations with application to load flow calculations in power systems”, Mathematical Modelling, 5, 1984, pp. 275-297.
[33] W. Sun, and Y. X. Yuan, “Optimization theory and methods: nonlinear programming”, Springer, 2006.
[34] M. N. Vrahatis, “Solving systems of nonlinear equations using the nonzero value of the topological degree”, ACM Transactions on Mathematical Software, 14, 1988, pp. 312-329
[35] M. N. Vrahatis, G.S. Androulakis, J. N. Lambrino, and G.D. Magoulas, “A class of gradient unconstrained minimization algorithms with adaptive stepsize”, Computer and Applied Math. 114, 2000, pp. 367–386.
[36] W. Wang, and Y. F. Xu, “A revised conjugate gradient projection algorithm for inequality constrained optimization”, J. of Computational Mathematics, 23, 2005, pp. 217-224
[37] Y. Xiao, and E. King-Wah Chu, “A nonmonotone inexact Newton algorithm for nonlinear systems of equations”, J. Austral. Math. Soc. Ser. B 36, 1995, pp. 460-492.
[38] F. Zhou, and Y. Xiao, “A class of nonmonotone stabilization Trust Region methods”, Computing 53, Springer-Verlag, 1994, pp.119-136

Nonlinear equations to solve, specified as a function handle or function name. fun is a function that accepts a vector x and returns a vector F, the nonlinear equations evaluated at x. The equations to solve are F = 0 for all components of F. The function fun can be specified as a function handle for a file.Â Internally, the 'levenberg-marquardt' algorithm uses an optimality tolerance (stopping criterion) of 1e-4 times FunctionTolerance and does not use OptimalityTolerance. OutputFcn. Specify one or more user-defined functions that an optimization function calls at each iteration.Â [8] Powell, M. J. D., â€œA Fortran Subroutine for Solving Systems of Nonlinear Algebraic Equations,â€ Numerical Methods for Nonlinear Algebraic Equations, P. Rabinowitz, ed., Ch.7, 1970. Extended Capabilities. Solving a system of nonlinear equations with the use of optimization methods in problems related to the wheel-rail contact. Article. Full-text available.Â In this paper, we present a global convergent derivative-free method that is capable to calculate all zeros using an appropriate Schauder base. The component functions of F are only assumed to be Lipschitz-continuous. Therefore, our method outperforms the classical counterparts. A globally convergent linearly constrained lagrangian method for nonlinear optimization. a dissertation submitted to the department of management science and engineering and the committee on graduate studies. of stanford university in partial fulfillment of the requirements. A system of nonlinear equations is a system of two or more equations in two or more variables containing at least one equation that is not linear. Recall that a linear equation can take the form [latex]Ax+By+C=0[/latex]. Any equation that cannot be written in this form in nonlinear. The substitution method we used for linear systems is the same method we will use for nonlinear systems.Â Solve for the remaining variable. Check your solutions in both equations. Example: Solving a System of Nonlinear Equations Representing a Parabola and a Line. Solve the system of equations. [latex]\begin{array}{l}x-y=-1\hfill \\ y={x}^{2}+1\hfill \end{array}[/latex].